

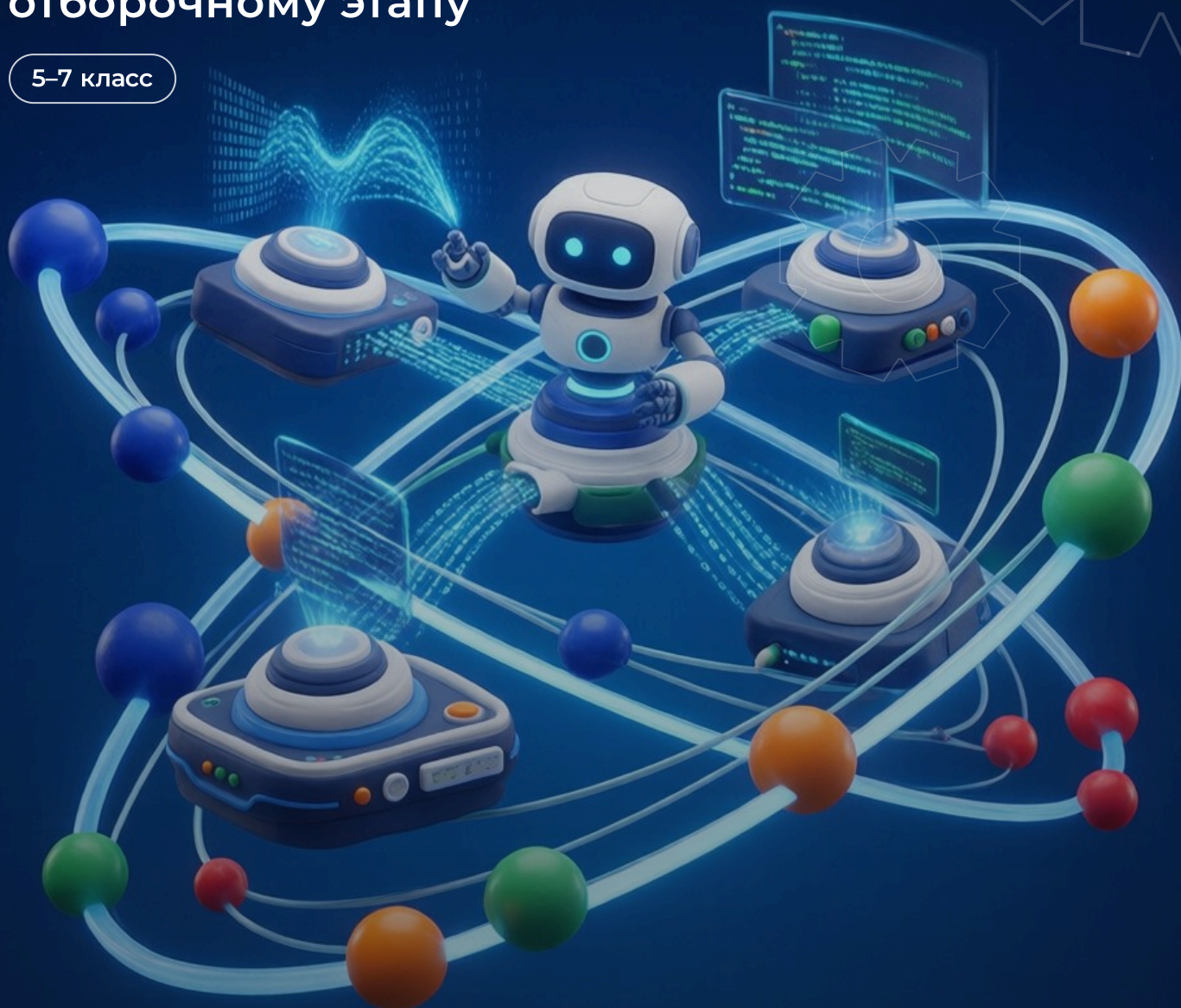


НТО ЮНИОРЫ 2026/27 · СФЕРА

Кибертех

Задачник-справочник для подготовки к
отборочному этапу

5–7 класс



Автор-составитель

Артём Осинowski, методист и разработчик детского технопарка «Кванториум»

Детский технопарк «Кванториум», Новосибирск · 2026

Кибертех

Задачник-справочник для подготовки к отборочному этапу НТО Юниоры. 5–7 класс

Автор-составитель – **Артём Осинковский**, методист и разработчик детского технопарка «Кванториум».

Нашли ошибку или неточность? Напишите на osinovsky.aa@kvantorium54.ru (рабочая почта): сфера, версия книги (1.0) и номер страницы – исправим в следующей версии. Актуальная версия и список изменений – nto.kvantorium54.ru/knigi. Личная почта для других вопросов – artem081033@gmail.com.

Важно. Это не официальное пособие олимпиады, а книга методиста «Кванториума». Каких заданий ждать на отборе 2026/27, мы не знаем: организаторы их заранее не показывают, а задания разных лет не повторяются. Книга готовит к темам из программы сферы и приёмам решения, а не к конкретным задачам.

Книга помогает школьнику самостоятельно подготовиться к отборочному этапу сферы «Кибертех» Национальной технологической олимпиады «НТО Юниоры»: теория с нуля, подробные разборы задач и задачи для самопроверки с ответами.

Большинство задач составлены автором по образовательной программе сферы, часть – по мотивам открытых заданий прошлых лет; источник указан у каждой задачи. Уровни сложности – шкала этой книги, а не баллы олимпиады. Правила, сроки и формат отбора – на сайте junior.ntcontest.ru.

Версия 1.0 от 02.10.2026, сборка 02.10.2026. © Артём Осинковский, 2026.

Содержание

Как устроена эта книга	4
Для кого эта книга	4
Как устроен отбор	4
Что пригодится на отборе и где это в книге	5
Как пользоваться книгой	6
Как вводить ответы на отборе	7
Что понадобится	7
План на месяц до отбора	16
Родителям и наставникам	16
Честно об этой книге	17
Глава 1 Что такое киберфизическая система	18
1.1. Кибер + физика	19
1.2. Цепочка сигнала и замкнутый круг	20
1.3. Порог: когда пора действовать	22
1.4. Время реакции: мир не ждёт	23
1.5. КФС вокруг нас	24
1.6. Связь и управление – две сквозные технологии	25
1.7. Турнир юных киберфизиков	26
1.8. Разбираем задачи	27
Глава 2 Сигнал и его параметры	39
2.1. Явление и сигнал	39
2.2. Аналоговый и цифровой сигнал	41
2.3. Колебания: амплитуда, период, частота, фаза	42
2.4. Дискретизация: сигнал по точкам	45
2.5. Детектирование: порог, период, шум	47
2.6. Разбираем задачи	49
Глава 3 Разложение сигнала: спектр и фильтры	57
3.1. Сложная волна – сумма простых	57
3.2. Спектр – «рецепт» сигнала	59
3.3. Шум и фильтр низких частот	61
3.4. Скользящее среднее	63
3.5. Несущая и огибающая	65
3.6. Аномалия: ищем лишнее	67
3.7. Разбираем задачи	68
Глава 4 Кодирование и системы счисления	78
4.1. Код, бит и байт	79
4.2. Двоичная система счисления	80
4.3. Шестнадцатеричная система	82
4.4. Скорость передачи	84
4.5. Как поймать ошибку: чётность и контрольная сумма	85
4.6. Азбука Морзе	87

4.7. Свой код без разделителей	88
4.8. Разбираем задачи	89
Глава 5 Модуляция и каналы связи	98
5.1. Зачем нужна модуляция	98
5.2. Амплитудная модуляция: громче – тише	100
5.3. Частотная модуляция: выше – ниже тон	101
5.4. Канал связи: звук и инфракрасный луч	104
5.5. Скорость, время и надёжность	106
5.6. Два канала сразу	107
5.7. Разбираем задачи	108
Глава 6 ПРИМС: машины состояний в Кибериаде	121
6.1. Состояние: устройство помнит, что с ним было	122
6.2. Событие, условие, действие, переход	123
6.3. Трассировка: идём за машиной шаг за шагом	125
6.4. Вложенные состояния	127
6.5. Ошибки в диаграмме: тупики, недостижимые состояния, путаница	128
6.6. Как это выглядит в Кибериаде IDE	130
6.7. Разбираем задачи	131
Глава 7 Управление КиберМишкой	144
7.1. Исполнитель и его команды	145
7.2. Алгоритм управления: схема из состояний	148
7.3. Данные датчика: числа, порог, события	150
7.4. Трассировка: ведём исполнителя пальцем	152
7.5. Разбираем задачи	153
Глава 8 Математика киберфизика	168
8.1. Формула – это фраза	168
8.2. Единицы: герцы, секунды, биты	170
8.3. Доли, проценты, пропорции	171
8.4. Уравнения и системы уравнений	172
8.5. Логика: И, ИЛИ, НЕ	174
8.6. Комбинаторика: сколько вариантов кода	176
8.7. Электронные таблицы	177
8.8. Разбираем задачи	178
Ответы и решения	188
Справочник	195
Числа, которые используются в книге	195
Перевод единиц	195
Формулы и правила	196
Мини-шпаргалки по инструментам	202
Словарик	205
Шпаргалка: всё «Запомни» из глав	208
Где что найти	214

Как устроена эта книга

Эта книга – тренажёр для отборочного этапа сферы «Кибертех» олимпиады «НТО Юниоры». Здесь есть всё, чтобы готовиться самостоятельно: теория с нуля, подробные разборы задач и сразу после каждого – похожие задачи «Реши сам» с ответами в конце книги. Покупать КиберМишку и паять ничего не придётся: большая часть задач решается в тетради, а остальное – в электронной таблице и в бесплатной Кибериаде IDE со встроенным симулятором.

Для кого эта книга

- Для школьников **5–7 классов**, которые выбрали (или думают выбрать) сферу «Кибертех».
- Для тех, кто **ещё не проходил** физику в школе и ни разу не программировал. Что такое частота, чем бит отличается от байта, зачем роботу «состояния» – всё объясняем с первого шага.
- Для **наставников и родителей**, которые хотят помочь. Для них есть отдельный раздел в конце введения.

Как устроен отбор

Таблица В.1

Когда	Что происходит
с 1 октября	демо-этап: пробные задания в личном кабинете
до 9 ноября	регистрация на олимпиаду (наш совет – успеть до 20 октября, тогда будут обе попытки)
20–30 октября	отбор, попытка 1 (3 часа)
31 октября – 10 ноября	отбор, попытка 2 (засчитывают лучшую из двух)
7–17 декабря	финалы: у каждой сферы свой день, один на всю страну

- Отбор проходит **дома, онлайн, с компьютера**. Задания открываются в личном кабинете на my.ntcontest.ru и решаются на платформе «Орбита» (orbita.education). Заходи туда только кнопкой из личного кабинета. Каждый решает сам за себя.
- Проверяет компьютер. Ответ – это число, строка из цифр выбранных вариантов или короткий код. На задачу даётся от трёх до пяти попыток ввода, и каждая неверная снимает 30 % балла за эту задачу. Ответ засчитан, только когда нажата кнопка «Отправить», а не «Сохранить».
- На вебинаре для наставников в сентябре 2026 года разработчики сферы объяснили, как всё связано. К отбору готовит первая часть курса: введение в киберфизику, физика

сигнала, разложение сигнала, кодирование и модуляция и два модуля про ПРИМС. Все они открываются до начала отбора. Задания и симуляторы на отборе те же, что в курсе: схему собирают в Кибериаде IDE с её симуляторами, а решение загружают в проверочную систему. Поэтому книга идёт по программе сферы.

- В курсе есть задачи, где сигнал приходит таблицей чисел. Его надо обработать в Excel или другой электронной таблице и ввести, что получилось. Таблицы есть и в списке того, что пригодится на отборе, так что без них не обойтись.
- Координатор сферы советует **пройти обе попытки**. В первой ты осваиваешься: где что нажимать, как выглядят задания. Во второй уже решаешь на результат. Засчитают лучшую, так что потерять нечего.
- За тест бесплатного курса своей сферы дают **+5 баллов** к отбору. Курс «Кибертех» – academy.junior.online/event/cybertech_junior, вход через «Талант». Бонус нужно включить самому в личном кабинете («Мои привилегии») до конца отбора. Учти две вещи. После того как бонус применён, сферу уже не поменять. И бонус засчитают, только если за задания ты набрал больше нуля.
- С 1 по 20 октября у сферы проходит открытый урок «Как найти аномалию в сообщении» (talent.kruzhok.org, материалы открываются после заявки). Тема близка к нашим главам 4 и 5.
- Финал – **командный** и очный. Главная часть, по которой составляют рейтинг, – задачи на компьютерах в симуляторах: команда настраивает передачу сообщения сразу по двум каналам. В регионах, где есть оборудование, добавляют «железный» хакатон с настоящими КиберМишками, в остальных – командную игру.

Совет. Сроки и правила могут уточняться. Главный источник – сайт олимпиады junior.ntcontest.ru и памятка для Новосибирской области nto.kvantorium54.ru.

Что пригодится на отборе и где это в книге

На странице сферы перечислено, что пригодится на отборе. Вот где это в книге:

Таблица В.2

Что пригодится на отборе	Главы и разделы
Как устроены киберфизические системы и датчики	1, раздел 7.3
Вычисления по формулам, величины и единицы	2, разделы 8.1–8.2
Работа с электронными таблицами	разделы 3.4 и 8.7
Системы счисления, обработка текста и картинок	4
Логика и комбинаторика: коды, биты, каналы	4, 5, разделы 8.5–8.6
Визуальная среда программирования, переменные и условия	6
Алгоритмы управления устройством	7
Уравнения, пропорции, дроби, проценты	разделы 8.3–8.4

Физики в этом списке нет, а сигнал – это физика: звук, свет, колебания. Поэтому всё физическое в главе 2 мы объясняем с нуля, на примерах из жизни.

Главу 1 прочитай первой: в ней появляются слова, на которых стоит вся книга. Главы 2, 3 и 5 идут цепочкой – сначала простой сигнал, потом сложный, потом как сигнал несёт сообщение. Главу 4 можно читать когда угодно, ей хватает тетради и карандаша. Главы 6 и 7 читай подряд и с компьютером: в них всё собирается в Кибериаде IDE. Глава 8 – общая математика для всех остальных; загляни в неё, как только споткнёшься о проценты или уравнение.

Как пользоваться книгой

В каждой главе теория и задачи идут вперемешку: после каждого разбора сразу стоят две похожие задачи «Реши сам». Приём закрепляется, пока он свежий, и не надо листать в конец главы.

1. **Теория.** Каждая новая тема начинается с большой картинки «на пальцах», потом идёт нормальное объяснение с формулами и рисунками. Самое нужное собрано в рамках «Запомни». Все «Запомни» из всех глав повторяются в конце книги, в шпаргалке.
2. **Разбираем задачи.** Задачи такого же вида, как на отборе, и к каждой – решение по шагам: что дано, что делаем и почему, где обычно ошибаются, какой ответ получился и можно ли ему верить.
3. **Реши сам.** Две задачи на тот же приём, но с другими числами и сюжетом. Вторая иногда с небольшим поворотом.
4. **Ответы** – в конце книги, в разделе «Ответы и решения». Там только ответ и иногда подсказка в одну строку, если в задаче есть подвох. Полного решения нет специально: если подсказки не хватило, вернись к разбору перед задачей.

Хочешь пробежаться по всей программе быстро – есть отдельная короткая книжка **«Быстрый старт»**: из каждой главы главное, один разбор и несколько задач «Реши сам». Её удобно пройти за неделю, а эту книгу держать под рукой как справочник.

У задач есть уровни, как в обычном задачнике. У трудных рядом с номером стоит звёздочка («задача 3*»): в них нужно несколько идей сразу, и они труднее, чем на отборе. Лёгкие, в одно действие, подписаны словом «разминка». Это уровни нашей книги, а не баллы олимпиады: сколько задач будет на отборе и сколько стоит каждая, заранее не объявляют.

Совет. Разбор читай с листком бумаги. Закрой решение, попробуй решить сам, а потом сверяйся по шагам. Так ты запомнишь в три раза больше, чем если просто прочитаешь.

У каждой задачи внизу написано, откуда она взялась. «Своя задача по программе сферы» – мы придумали её по теме из программы 2026/27 года. «По мотивам задачи» – идею мы взяли из настоящей олимпиадной задачи, пересказали своими словами и поменяли числа. Задач прошлых лет у «Кибертех» почти нет в открытом доступе, поэтому своих задач в книге большинство. После демо-этапа мы сверим их с настоящим форматом отбора.

Как вводить ответы на отборе

Так было на отборочных этапах прошлых лет в других сферах. На «Орбите» перед каждой задачей читай инструкцию: там сказано, в каком виде нужен ответ.

- **Только число**, без единиц измерения. Смотри, в чём просят ответ: в герцах или килогерцах, в секундах или миллисекундах, в битах или байтах. Перепутать миллисекунды с секундами – верный способ потерять баллы за задачу, которую ты решил правильно.
- **Дробную часть** на сайте обычно отделяют **точкой**: `0.25`, а не `0,25`. В книге мы пишем по-русски, через запятую.
- **Двоичный или шестнадцатеричный код** вводи без пробелов и без приписок вроде `0b` или `0x`, если условие не просит иначе. Посмотри, нужны ли ведущие нули: код `00101101` и число `101101` – для проверяющей программы разные строки.
- **Номера вариантов** записывай подряд, без пробелов и запятых, если не сказано по-другому: `134`.
- **Округляй так, как сказано в условии**. Промежуточные результаты не округляй – только ответ.
- **Сколько бит нужно** – всегда целое число и всегда с округлением вверх. На 33 буквы пяти бит не хватит (это только 32 кода), нужно шесть.
- **Попытки ограничены**. Сначала реши на бумаге, проверь вычисления в таблице или схему в симуляторе Кибериады – и только потом вводи ответ.

Что понадобится

- Тетрадь в клетку, карандаш, линейка – графики сигналов удобно рисовать по клеточкам.
- Калькулятор. На компьютере с Windows есть встроенный калькулятор с режимом «Программист»: он сам переводит числа в двоичную и шестнадцатеричную системы. Им удобно проверять себя, но переводить научись и вручную – глава 4.
- Компьютер с интернетом. Для Кибериады IDE нужен компьютер, не планшет и не телефон: есть версии для Windows, Linux и macOS.
- **Кибериада IDE** – среда, где программу для КиберМишки рисуют схемой из состояний и стрелок. В неё встроен симулятор, так что настоящий КиберМишка для подготовки не нужен. Как поставить – в справочнике в конце книги.
- **Электронная таблица** – Excel или бесплатный LibreOffice Calc. В ней строят график сигнала и сглаживают его.
- **Python** – по желанию. Разработчики сферы говорят, что он не обязателен, но длинные ряды чисел в нём обрабатывать быстрее, чем в таблице. Хватит любого онлайн-интерпретатора.

Кибериада IDE: установка и первая программа

Кибериада IDE – бесплатная отечественная среда, в которой программу не пишут строчками, а рисуют диаграммой на языке ПРИМС: режимы работы – прямоугольники-состояния, смена режима – стрелки-переходы. Ниже весь путь от установки до первой программы «мигалка» – задача из курса сферы. Рамка с номером на картинке – куда нажимать.

Шаг 1. Скачать и поставить. Открой method.insitulab.pro/cyberiada и нажми «Скачать программу». Выбери файл под свою систему: `.exe` – Windows, `.dmg` – macOS, `.deb` или `.AppImage` – Linux. Запусти его. Установщик попросит права администратора – он ставит драйверы для плат, так и должно быть; дома нажимай «Да» вместе со взрослым. После установки программа откроется сама.

Шаг 2. Новый документ. Слева – панель «Документ», в ней «Создать...» (1).



Рисунок В.1 – Стартовый экран Кибериады: «Документ» → «Создать...»

Шаг 3. Платформа. Кибериада умеет работать с разными платами, поэтому сначала выбирают, для какой пишем. Дважды щёлкни «Arduino Uno» в списке платформ (1) – она появится в «Выбрано» (2). Нажми «Создать» (3). В Кибертехе задачи курса решают в задачнике Кибериады на роботе «Садовник» (вход – из курса сферы, модуль 5): там свои компоненты – Движение, Цветы, Таймер, Сенсор, Компас, – а таймер срабатывает один раз, и его запускают заново. Порядок работы с диаграммой тот же, что ниже.

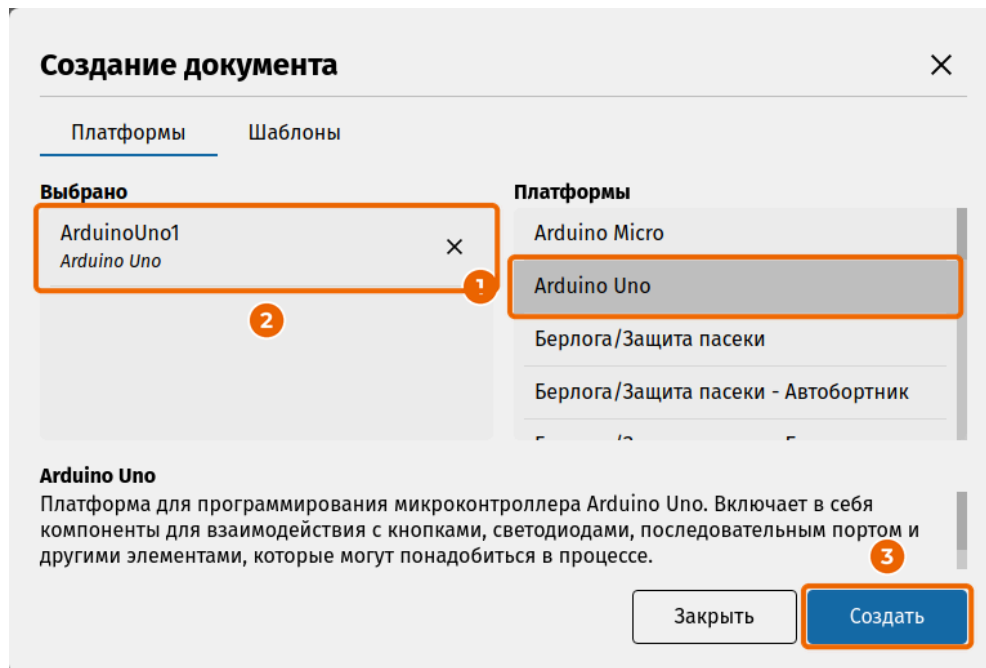


Рисунок В.2 – Выбор платформы для документа

Шаг 4. Компоненты. Компонент – это то, чем программа управляет или от чего ждёт сигнала: светодиод, кнопка, таймер. Нажми «+» у «Компонентов» (1).

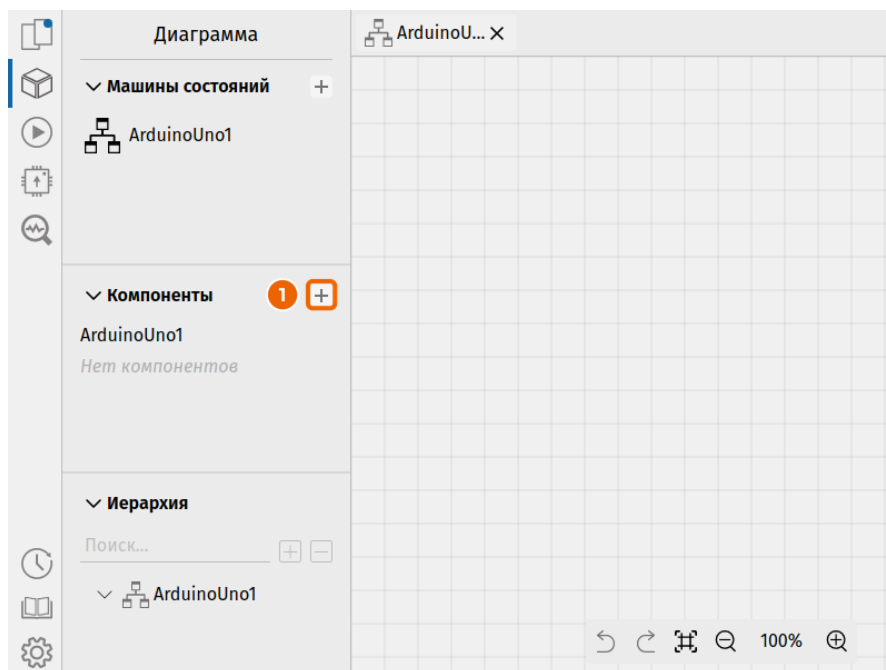


Рисунок В.3 – Окно диаграммы: плюс у «Компонентов»

Выбери «Светодиод» (1 на левой половине) и нажми «Добавить» (2). В форме впиши название «Светодиод» (1 на правой половине) и выбери пин (2) – номер вывода Arduino, к которому подключён светодиод; на занятии курса это 2. Нажми «Применить» (3). Так же добавь «Таймер» – у него пина нет, только название «Таймер».

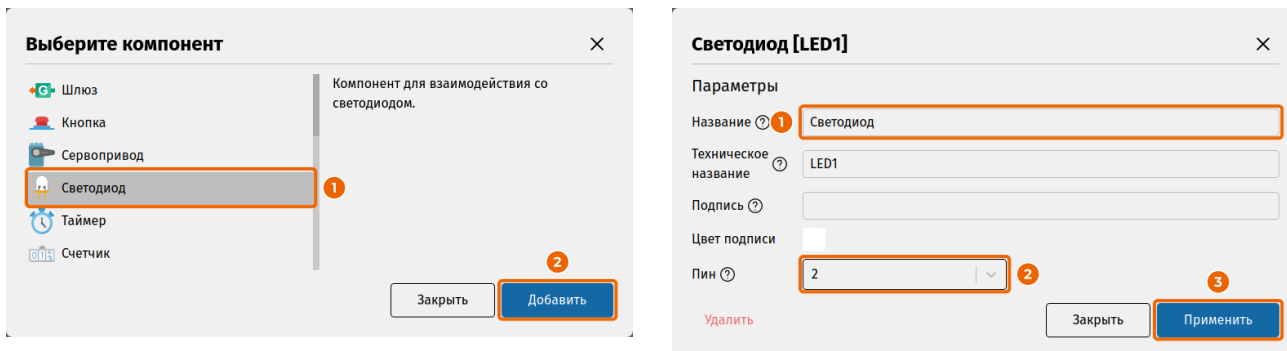


Рисунок В.4 – Новый компонент: слева – выбор «Светодиода», справа – его название и пин

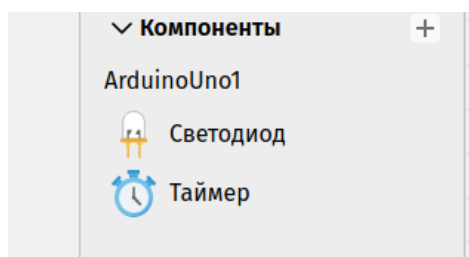


Рисунок В.5 – Оба компонента в списке

Шаг 5. Состояние «Не горит». Дважды щёлкни по пустому месту поля – появится прямоугольник «Состояние». К нему от синего кружка ведёт стрелка: отсюда программа начинает работу. Нажми карандаш (1) и впиши «Не горит», затем Enter.

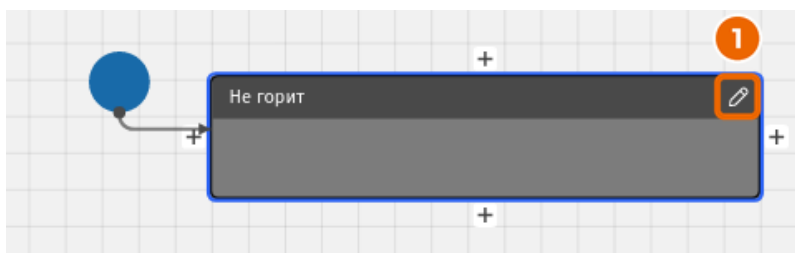


Рисунок В.6 – Новое состояние: карандаш – переименовать

Теперь скажем, что делать в этом состоянии. Дважды щёлкни по нему – откроется «Редактор состояния». Нажми «+» (1).

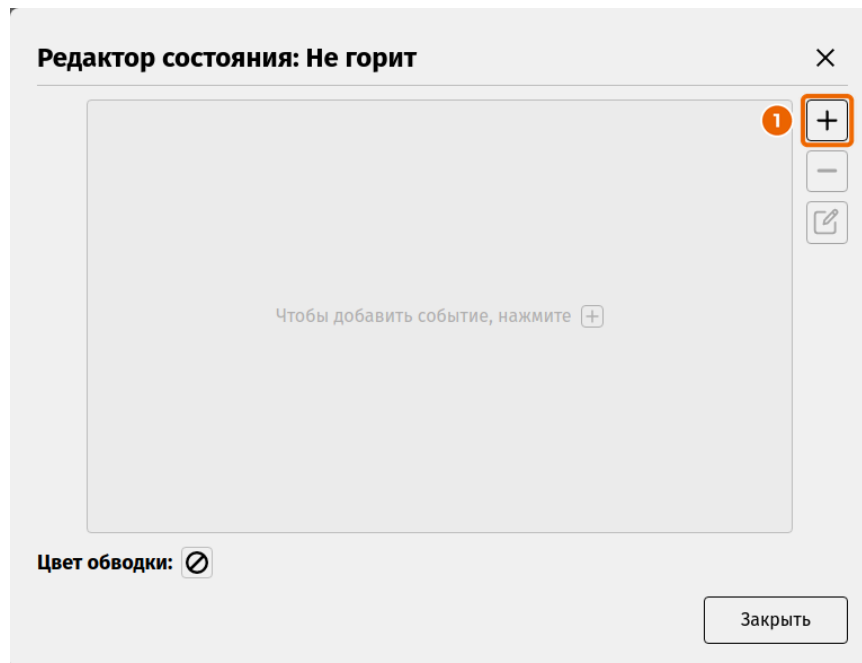


Рисунок В.7 – Редактор состояния

В «Редакторе события» в строке «Когда» уже стоит «Общие → Вход» (1): действия выполняются, как только программа войдёт в это состояние. Так и оставь. Нажми «+» у «Делай» (2).

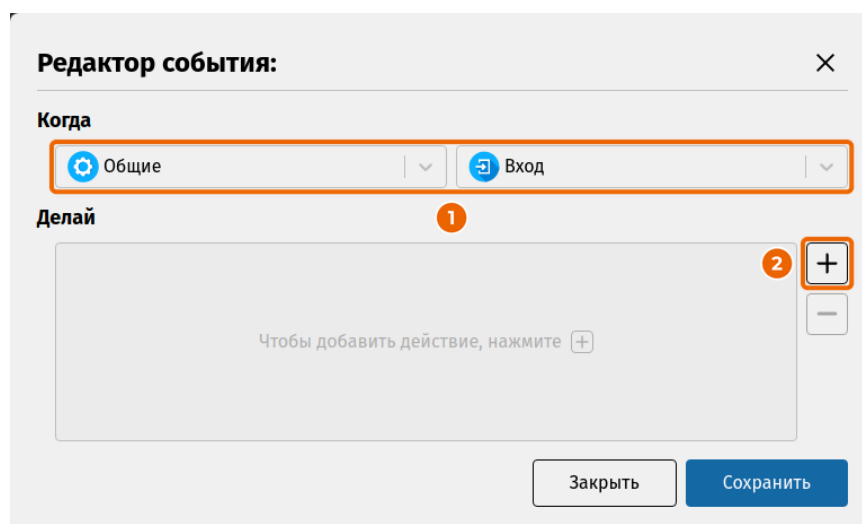


Рисунок В.8 – Редактор события: «Вход» и плюс у «Делай»

Выбери компонент «Светодиод» (1) и действие «Погасить» (2), нажми «Сохранить».

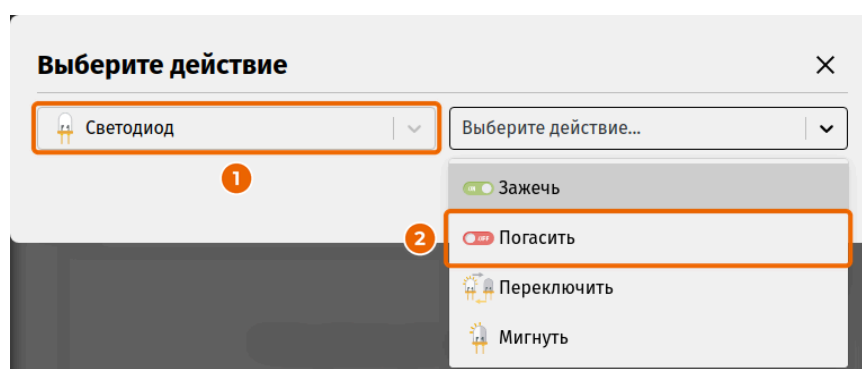


Рисунок В.9 – Действие «Светодиод – Погасить»

Ещё раз «+» у «Делай»: компонент «Таймер» (1), действие «Старт» (2), в поле интервала – 1000 (3), это миллисекунды, то есть одна секунда. Нажми «Сохранить» (4). Переключатель «Интервал» не трогай: он нужен, чтобы брать число из другого компонента, а нам хватит простого числа.

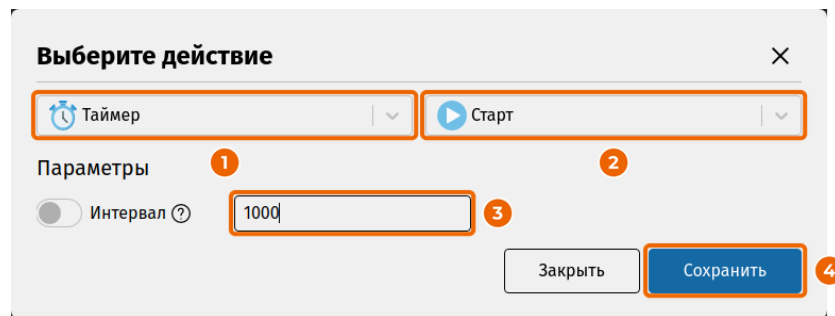


Рисунок В.10 – Действие «Таймер – Старт» на 1000 мс

В «Делай» теперь два действия (1). Нажми «Сохранить» (2), потом «Заккрыть» в редакторе состояния.

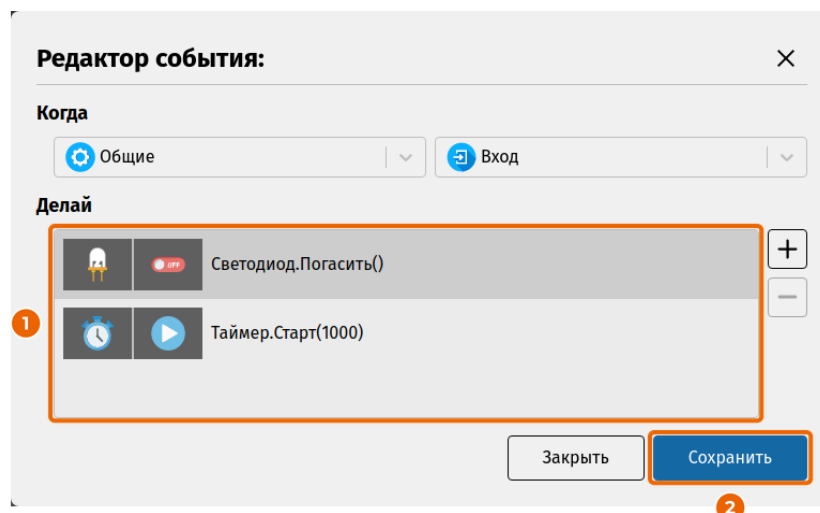


Рисунок В.11 – Событие «Вход» с двумя действиями

Шаг 6. Состояние «Горит». Так же: двойной щелчок по пустому месту, имя «Горит», одно действие – «Светодиод → Зажечь». Таймер второй раз запускать не нужно: на платформе Arduino Uno таймер повторяющийся – запустили один раз, и он срабатывает каждую секунду. На других платформах бывает иначе – смотри описание компонента.

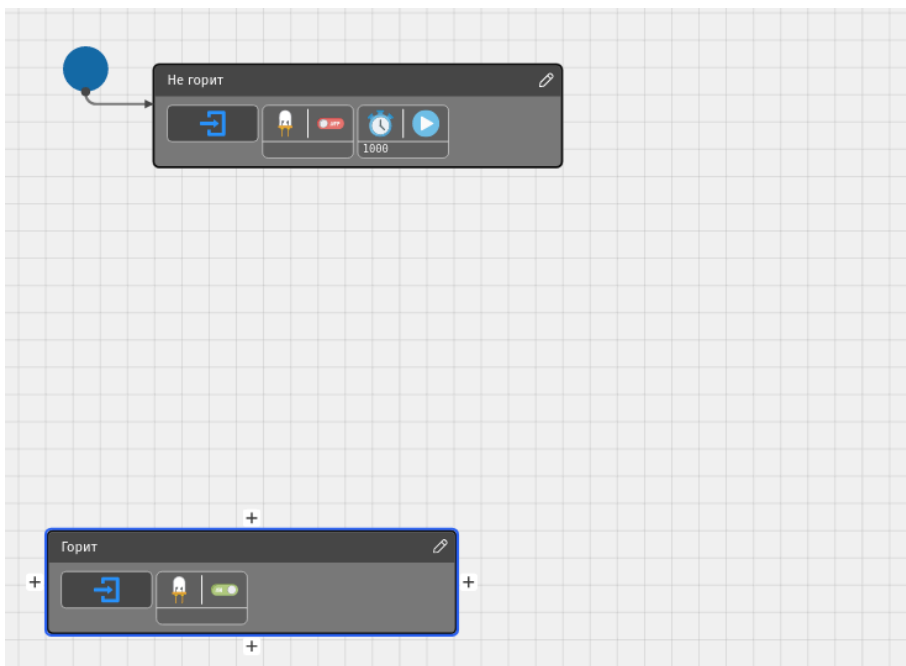


Рисунок В.12 – Два состояния готовы: на каждом видно, что оно делает при входе

Шаг 7. Переходы. Щёлкни по «Не горит» – вокруг появятся плюсики. Потяни мышью нижний плюстик (1) до состояния «Горит» (2) и отпусти.

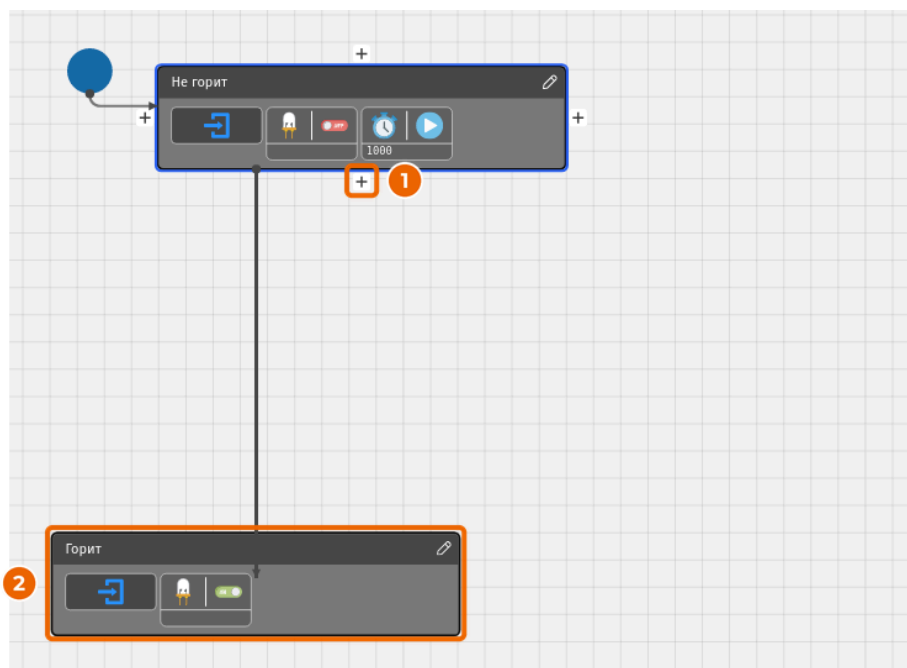


Рисунок В.13 – Тянем стрелку перехода

Откроется «Редактор соединения». В «Когда» выбери «Таймер» (1) и «Тайм-аут» (2) – переход случится, когда таймер отсчитает секунду. Условия и действия не нужны. Нажми «Сохранить» (3).

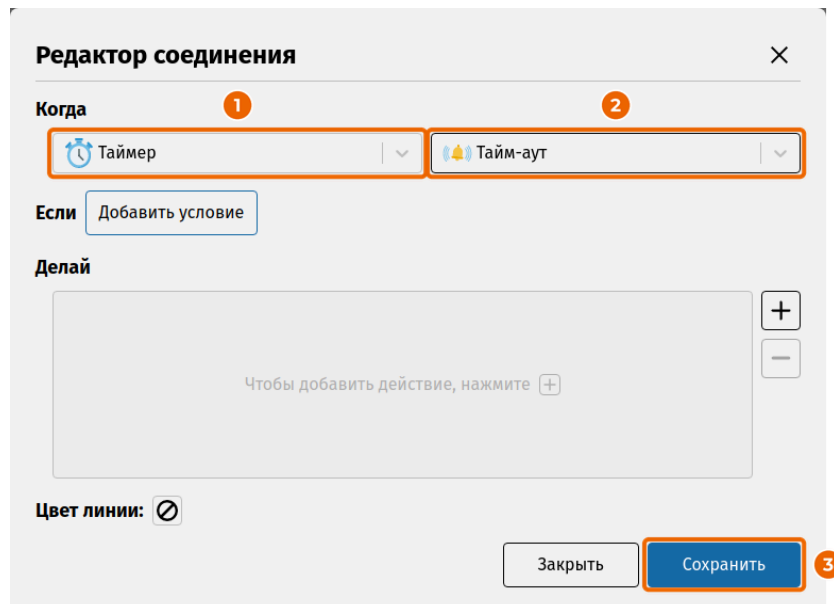


Рисунок В.14 – Переход по тайм-ауту таймера

Так же протяни стрелку обратно – из «Горит» в «Не горит», с тем же событием. Если две стрелки легли одна на другую, перетащи значок перехода мышью в сторону.

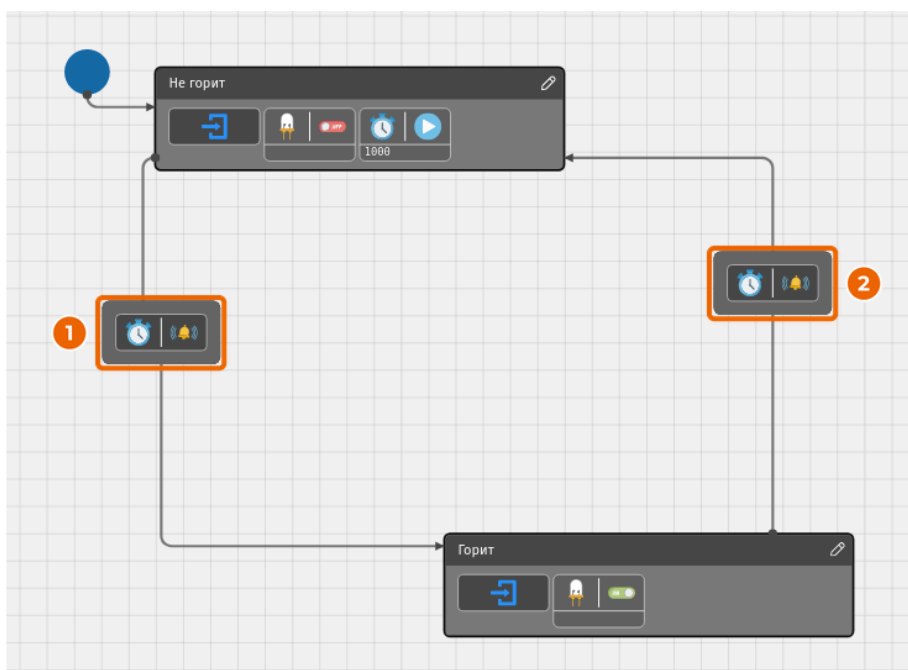


Рисунок В.15 – Готовая «мигалка»: переход «Не горит → Горит» (1) и обратный (2)

Прочитай программу по стрелкам: старт → «Не горит» (погасить, запустить таймер) → через секунду тайм-аут → «Горит» (зажечь) → тайм-аут → «Не горит» – и так по кругу. Светодиод мигает раз в секунду. Сохрани работу: «Документ → Сохранить как...», программа сохранится файлом `.graphml`.

Карта значков слева. По ним переключают панели.

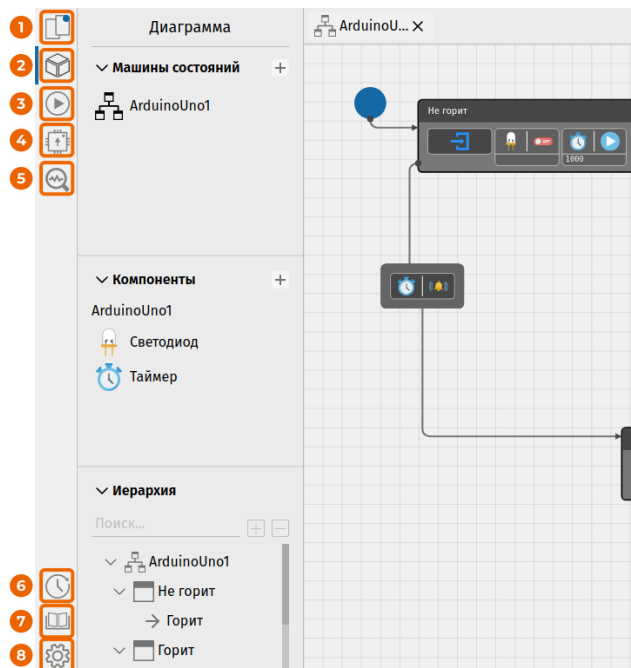


Рисунок В.16 – Значки левой панели

1. **Документ** – создать, открыть, сохранить.
2. **Диаграмма** – машины состояний, компоненты, иерархия состояний.
3. **Компилятор** – превращает диаграмму в прошивку для платы.
4. **Загрузчик** – записывает прошивку в плату через USB.
5. **Монитор порта** – показывает, что плата присылает по USB.
6. **История изменений** – что и когда менялось в документе.
7. **Документация** – справка (её же открывает F1).
8. **Настройки**.

Шаг 8. Компиляция и прошивка – когда есть плата. Открой «Компилятор» (3 на карте) и нажми большую синюю кнопку (1) – «Скомпилировать». Кружок рядом с заголовком (2) показывает связь с компилятором: если он красный, кнопка называется «Переподключиться» – нажми её и дождись, пока кружок станет зелёным.

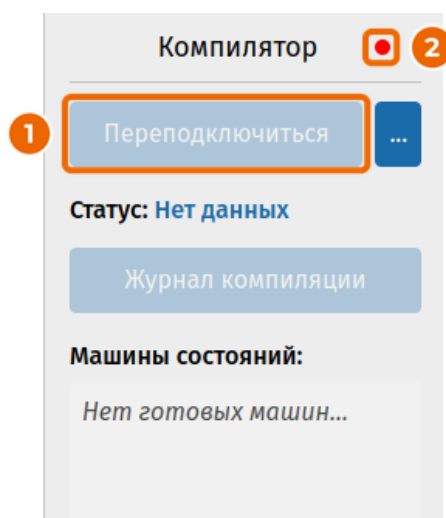


Рисунок В.17 – Панель «Компилятор»

Подключи Arduino к компьютеру кабелем и открой «Загрузчик» (4 на карте). Нажми «Подключить плату» (1), выбери свою плату в списке и нажми синюю кнопку с изображением платы – «Прошить». Прошивка готова, когда в «Журнале действий» появится подтверждение.

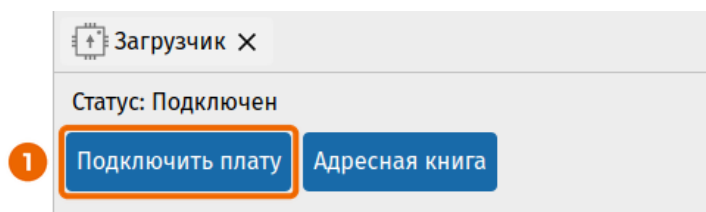


Рисунок В.18 – Панель «Загрузчик»

Платы под рукой нет – диаграмму всё равно можно проверить: пройди пальцем по стрелкам, как в разборах этой книги, и посмотри, в каком состоянии окажется программа после каждого события.

План на месяц до отбора

Если начинаешь в конце сентября, вот спокойный план: примерно по часу в день.

- **Неделя 1 – главы 1–3.** Записаться на курс сферы, решить демо-задания, построить в таблице график сигнала и сгладить его.
- **Неделя 2 – главы 4–5.** Каждый день 3–4 задачи на переводы чисел и коды. Поставить Кибериаду IDE.
- **Неделя 3 – главы 6–7.** Собрать машины состояний из разборов и прогнать их в симуляторе.
- **Неделя 4 – глава 8 и шпаргалка.** Заново решить задачи, где были ошибки. Пройти тест курса сферы (+5 баллов).

Отбор длится до 10 ноября, и попыток две. Если первая прошла неудачно, вернись к главам, где потерял баллы, и иди на вторую.

Родителям и наставникам

- **Помогайте разбираться, но не решайте за ребёнка.** Во время попытки подсказывать нельзя, иначе дисквалификация. А на командном финале сразу видно, кто разобрался сам. Лучше пусть ребёнок ошибётся на отборе и научится, чем попадёт в финал и растеряется там.
- **Поставьте Кибериаду IDE вместе, заранее.** При установке Windows может спросить разрешение на запуск и права администратора для драйверов. Ребёнку в одиночку это непросто, а в день отбора на это не должно уходить время.
- **Садитесь рядом на первых задачах курса.** Детей чаще всего останавливает не сложность, а «не ставится», «что-то зависло» или «непонятно, куда нажать». Разработчики честно говорят: курс местами сложнее школьной программы, и ребёнок сначала поймёт не всё. Ваша задача – подбодрить и предложить прочитать ещё раз, а не объяснить всё за него.
- **Чат сферы в MAX – для детей.** Разработчики просят взрослых писать не туда, а в отдельный контакт для родителей и наставников (ссылка goo.su/JY6uN из вводной пре-

зентации сферы).

- **Всё про регистрацию, бонусы и финал** для Новосибирской области – на nto.kvantorium54.ru (вкладки «Я наставник» и «Я участник или родитель»).
- **Региональный координатор НТО Юниоры в Новосибирской области** – детский технопарк «Кванториум», info@kvantorium54.ru.

Честно об этой книге

Главное, что нужно знать про эту книгу: заданий отбора 2026/27 мы не видели – организаторы их заранее не показывают. Угадывать их бесполезно: наставники, которые следят за олимпиадой не первый год, говорят, что задания из года в год не повторяются. Поэтому книга учит темам из программы сферы и приёмам решения, а не готовит к конкретным задачам. Когда организаторы выложат пробные задания, мы сверим с ними книгу и выпустим новую версию.

Это не официальное издание НТО. Книгу составил методист «Кванториума» по открытым материалам олимпиады: программе кружка «Киберфизика. Основы сигналов и управления» 2026/27, записям вебинаров сферы, инструкции к Кибериаде IDE и сборникам задач старших профилей НТО «Технологии беспроводной связи» и «Интеллектуальные энергетические системы». Задачи оттуда рассчитаны на 8–11 класс, поэтому мы брали только идеи и сильно упрощали.

«Кибертех» – новое имя сферы, которая в прошлом сезоне называлась «Технологии и киберфизические системы». Задания её отбора лежат на «Орбите» в закрытом архиве, и увидеть их можно только участникам того сезона. Поэтому большинство задач здесь наши, по программе сферы. С 1 октября откроется демо-этап – после него мы сверим книгу с настоящим форматом заданий и поправим, если что-то не совпадёт.

Кибериада IDE обновляется, а кнопки и окна от версии к версии могут меняться. Поэтому интерфейс мы описываем в общих чертах, а смысл – подробно: состояние, событие и переход останутся теми же. Точную инструкцию к своей версии смотри в справке программы (клавиша F1) и в курсе сферы.

Каждое число в ответах мы пересчитали на компьютере, а каждую программу запустили. Если ты всё-таки нашёл ошибку или что-то осталось непонятным, напиши автору: osinovsky.aa@kvantorium54.ru. Следующие версии книги станут лучше благодаря тебе.

Что такое киберфизическая система



Фонарь у подъезда загорается, когда стемнеет. Автобус тормозит, если перед ним выскочила собака. Спутник сам поворачивает солнечные панели к Солнцу. Человек в этих историях не нужен: устройство само замечает перемену и само на неё отвечает. В главе – из каких частей собраны такие системы, как по ним бежит сигнал и сколько метров успевают проехать машина, пока автоматика думает. Формул почти нет. Зато много задач «разложи по полочкам»: найти роль каждой детали, расставить события по порядку, выбрать верные утверждения. Ответ – число или строка из цифр.

1.1. Кибер + физика

НА ПАЛЬЦАХ

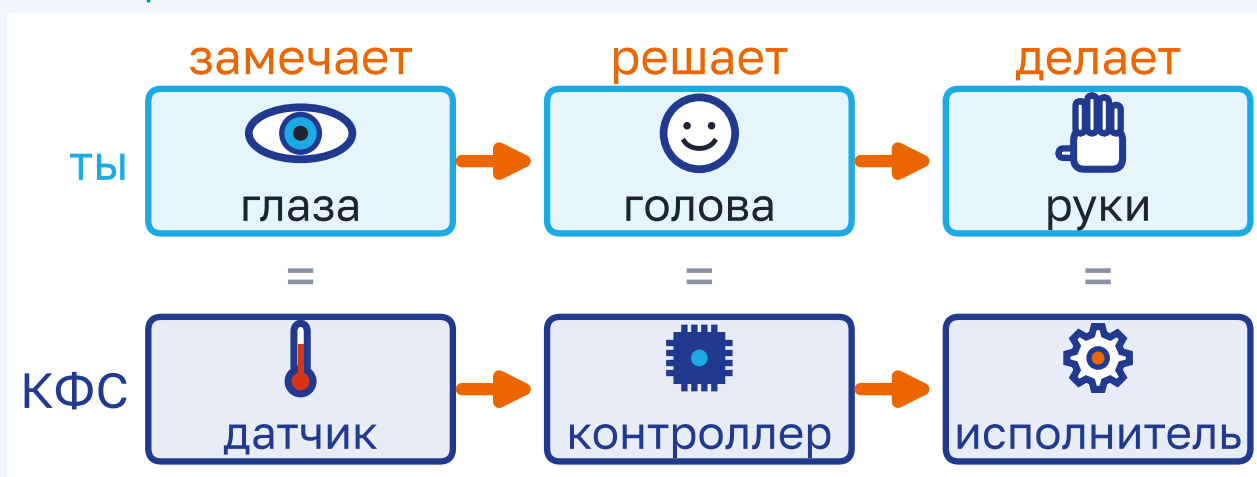


Рисунок 1.1 – Ты и автомат устроены похоже: есть чем замечать, чем решать и чем делать

Глаза – датчик, голова – контроллер, руки и ноги – исполнители. У автомата та же схема, только из проводов и моторов.

Посмотри, как ты переходишь дорогу. Глаза видят, что загорелся зелёный. Голова решает: «Можно идти». Ноги делают шаг. А если из-за поворота вылетел самокат, глаза это замечают, голова передумывает, ноги останавливаются. Ты всё время смотришь на мир и что-то в нём меняешь.

Киберфизическая система (сокращённо КФС) – устройство или набор устройств, которые делают то же самое без человека: сами измеряют, что происходит вокруг, сами выбирают, как поступить, и сами действуют в настоящем, физическом мире. Слово составлено из двух половинок. «Физическая» – потому что система работает с настоящими вещами: теплом, светом, водой, колёсами. «Кибер» – от слова «кибернетика», науки об управлении и связи; откуда оно взялось, узнаешь в разделе 1.6.

У любой КФС есть три главные роли:

- **датчик** – «органы чувств» системы. Он замечает, что происходит в мире (свет, тепло, звук, движение, влажность), и превращает это в **сигнал** – изменение напряжения или число, которое может прочитать электроника;
- **контроллер** – маленький компьютер, «голова» системы. В нём записана **прошивка** – программа, которая говорит, что делать с показаниями датчиков. Контроллер сравнивает, считает и принимает решение;
- **исполнитель** – «руки» системы. Он меняет что-то в мире: мотор крутит колесо, клапан открывает воду, лампа загорается, динамик пищит.

Чем КФС отличается от простого автомата? Сравни два уличных фонаря. Первый включается каждый день в 17:00 по часам. В декабре в Новосибирске в это время уже темно, всё хорошо. А в июне он зажжётся при ярком солнце и будет гореть зря. Второй фонарь смотрит на небо через датчик освещённости и включается, когда стало темно, – в любой месяц, в грозу, в солнечное затмение. Первый работает вслепую по заранее заданному плану. Второй **замечает**,

что происходит сейчас, и подстраивается под это. Второй – киберфизическая система, первый – нет.

Запомни. В КФС всегда есть три роли: датчик, контроллер с прошивкой и исполнитель, и работают они с настоящим миром. Без датчиков система не узнает, что в мире что-то поменялось. Без исполнителей – узнает, но ничего не сможет с этим сделать.

Совет. Чтобы найти в устройстве датчик, спроси: «Откуда оно узнаёт, что пора действовать?»
Чтобы найти исполнитель: «Что в мире сдвинется, загорится или зазвучит, когда оно решит?»

1.2. Цепочка сигнала и замкнутый круг

НА ПАЛЬЦАХ

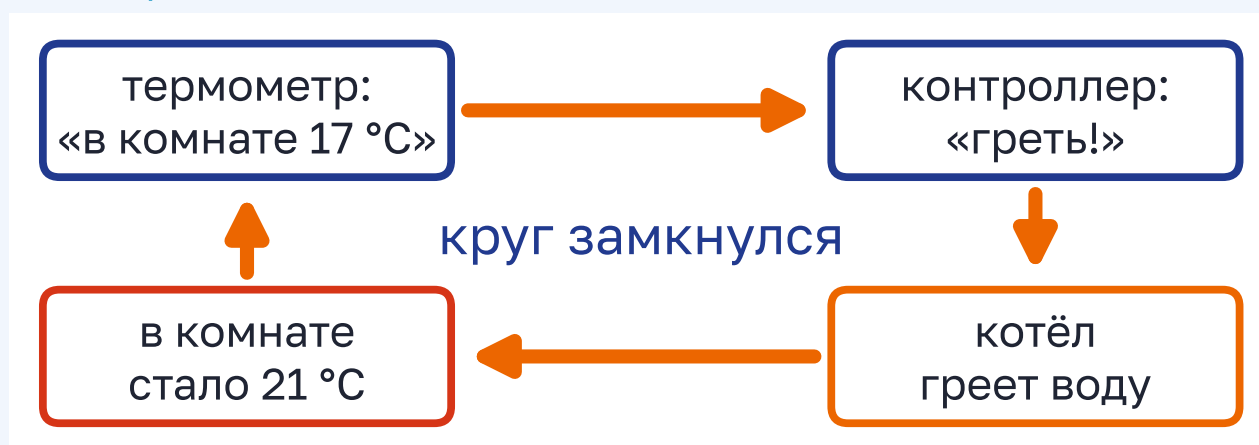


Рисунок 1.2 – Дача зимой: котёл греет дом, пока термометр не скажет «хватит»

Котёл нагрел комнату – термометр это заметил – контроллер выключил котёл.
Остыло – всё сначала. Круг замыкается через сам дом.

От датчика до исполнителя сигнал проходит пять звеньев, одно за другим.



Рисунок 1.3 – Пять звеньев КФС. Синие стрелки – информационные сигналы (что происходит в мире), оранжевые – управляющие (что делать). Круг замыкается через физический мир

1. **Датчик** измеряет величину: температуру, освещённость, расстояние до препятствия.

2. **Передача.** Показание нужно доставить до контроллера: по проводу, по радио (Wi-Fi, Bluetooth), иногда звуком или светом. Датчик может стоять в теплице, а контроллер – в доме в двадцати метрах.
3. **Обработка.** Контроллер читает число и сравнивает его с правилом из прошивки: «влажность почвы меньше 30 %?», «до стены меньше 20 см?».
4. **Управление.** По итогам сравнения контроллер отправляет команду: «открыть клапан», «остановить моторы».
5. **Исполнитель** выполняет команду и меняет мир.

Сигналы в этой цепочке бывают двух сортов. **Информационный сигнал** рассказывает, что происходит: «в комнате 17 °С», «впереди препятствие». Он идёт от датчиков к контроллеру. **Управляющий сигнал** – это команда: «включить нагрев», «тормозить». Он идёт от контроллера к исполнителям. Если про сигнал можно сказать «это новость», он информационный. Если «это приказ» – управляющий.

Замкнутый круг. Исполнитель поменял мир – и датчик это заметил. Котёл нагрел воздух, термометр показал 21 °С, контроллер выключил котёл. Такой круг называют **обратной связью**: результат действия возвращается в систему и влияет на следующее решение. Круг крутится всё время, пока система включена. Контроллер опрашивает датчик снова и снова: десять раз в секунду или раз в минуту – как задумал инженер.

Не путай. В курсе слова «пять звеньев» встречаются и в другой задаче: там цепочка идёт от события до человека: явление (дым поднимается к потолку) → датчик (превращает дым в напряжение) → канал (провод) → плата (разбирает напряжение и решает «дым есть») → получатель (охранник видит красную лампу). Это те же «датчик – передача – обработка», только с обоих концов добавлены явление и человек, а отдельных «управления» и «исполнителя» в ней нет. Сколько звеньев спрашивают, смотри по условию: какую цепочку нарисовали, такую и считай. Ещё одно слово курса: «обратной связью» там называют и пунктирную стрелку, по которой устройство сообщает плате, что команду выполнило.

Запомни. Звенья идут так: датчик → передача → обработка → управление → исполнитель, а через мир – снова датчик. Всё, что едет к контроллеру, – информационные сигналы. Всё, что уходит от него к исполнителям, – управляющие.

1.3. Порог: когда пора действовать

НА ПАЛЬЦАХ

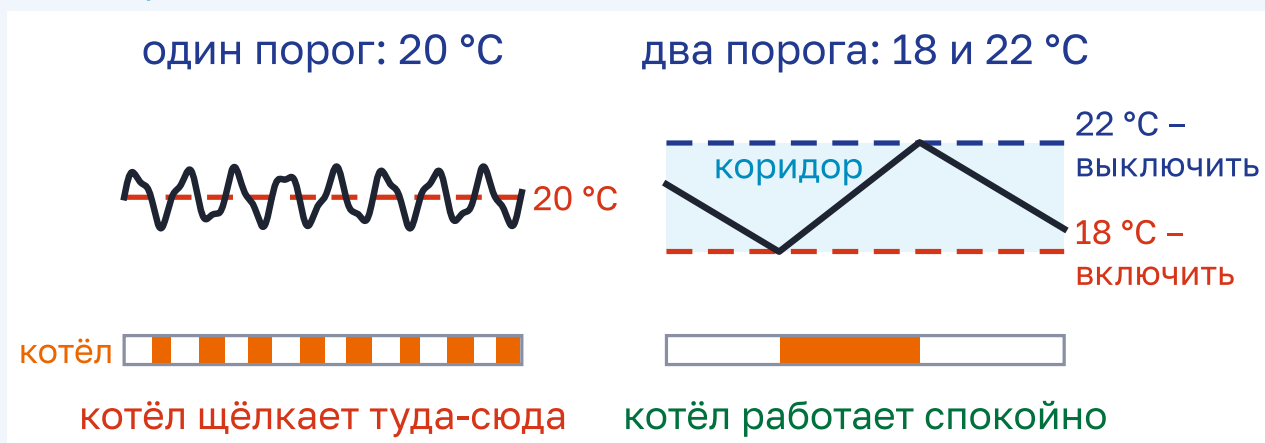


Рисунок 1.4 – Один порог и два порога. Оранжевая полоска – котёл включён

С одним порогом котёл дёргается без передышки. С двумя между ними появляется коридор, и котёл спокойно делает то, что делал.

Как контроллер понимает, что пора действовать? В простых системах он сравнивает показание датчика с заранее выбранным числом – **порогом**. Меньше порога – одно действие, больше – другое. Влажность почвы ниже 30 % – поливать, выше – не поливать.

С одним порогом бывает беда. Пусть котёл включается ниже 20 °C и выключается выше 20 °C. Температура в доме на месте не стоит: чуть поднялась – котёл выключился, чуть остыла – включился. Около двадцати градусов он будет щёлкать туда-сюда каждые несколько секунд и быстро сломается.

Посмотри, как поступаешь ты сам. Шапку ты надеваешь, когда замёрз, а снимаешь, когда стало жарко. Между «замёрз» и «жарко» ты её не трогаешь, иначе снимал бы и надевал на каждом шагу. С котлом делают так же: ставят **два порога**. Ниже 18 °C – включить, выше 22 °C – выключить, а между ними ничего не менять.

Тут есть хитрость. Пусть в доме 20 °C. Котёл сейчас может быть включён (дом греется после холода), а может быть выключен (дом остывает после жары). По одному термометру этого не узнать. Поэтому контроллер должен **помнить**, в каком он состоянии: «грею» или «не грею». Системы, которые помнят своё состояние, – тема глав 6 и 7.

Запомни. Порог – число, с которым контроллер сравнивает показание датчика. Два порога дают коридор: пока показание внутри него, система ничего не переключает и держит прежнее состояние.

Совет. Читай слова в условии по одному. «Ниже 12 °C» – это всё, что меньше 12, а сами 12 °C сюда не входят. «Не выше 16 °C» – наоборот, 16 входит. Маленькое «не» меняет ответ.

1.4. Время реакции: мир не ждёт

НА ПАЛЬЦАХ

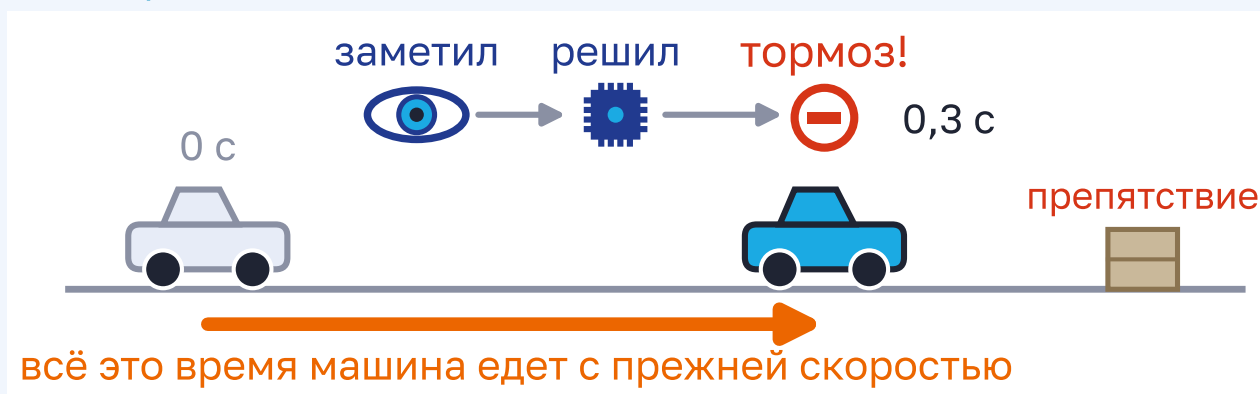


Рисунок 1.5 – Автоматика ещё думает, а машина уже едет дальше

Каждое звено тратит свою долю секунды. Складываем их – и узнаём, сколько метров машина проехала «вслепую».

Ни одно звено не срабатывает мгновенно. Датчик не смотрит каждое мгновение: его опрашивают через равные промежутки времени. Такой промежуток называют **периодом опроса**. Если датчик опрашивают 10 раз в секунду, период опроса $1 : 10 = 0,1$ с. Передача занимает время, обработка тоже, а тормозу или мотору нужно раскататься.

Время от события до действия называют **временем реакции**. Оно равно сумме всех задержек по дороге:

$$t_{\text{реакции}} = t_{\text{ожидания опроса}} + t_{\text{передачи}} + t_{\text{обработки}} + t_{\text{исполнителя}}.$$

Самый неудачный случай – событие случилось сразу после опроса. Тогда система узнает о нём только на следующем опросе, через целый период. Кто отвечает за безопасность, считает по этому, худшему, варианту: если автоматика успевает в худшем случае, она успеет и в любом другом.

Пока система реагирует, машина едет. Путь равен скорости, умноженной на время:

$$s = v \cdot t,$$

где s – путь (метры), v – скорость (метры в секунду), t – время (секунды). Скорость машины пишут в км/ч, а задержки – в секундах. Переводим: 1 км/ч – это 1000 м за 3600 с, то есть $1000 : 3600 = 1 : 3,6$ м/с. Отсюда правило: **чтобы перевести км/ч в м/с, дели на 3,6**. 36 км/ч = 10 м/с, 72 км/ч = 20 м/с.

Запомни. Время реакции – сумма всех задержек от события до действия. Считают его по худшему случаю: ожидание опроса – целый период. Путь за это время – скорость в м/с, умноженная на время в секундах.

1.5. КФС вокруг нас

НА ПАЛЬЦАХ



Рисунок 1.6 – Четыре киберфизические системы: у каждой есть чем заметить и чем сделать

Дом, перекрёсток, машина, спутник – размеры разные, схема одна.

Вот четыре системы, от дачи до орбиты.

Таблица 1.1

Система	Датчик	Решает	Исполнитель
Умный дом	термометр, датчик протечки	контроллер дома	котёл, кран на трубе
Светофор с кнопкой	кнопка, датчик машин	контроллер в шкафу	лампы светофора
Автопилот машины	камеры, радар	бортовой компьютер	руль, тормоз, мотор
Спутник	датчик Солнца	бортовой компьютер	маховики, моторы панелей

Умный дом. На даче зимой в доме никто не живёт, но трубы не должны замёрзнуть. Термометр в комнате меряет температуру, контроллер держит её между порогами и включает котёл. Датчик протечки лежит на полу у стиральной машины: если на него попала вода, контроллер перекрывает кран на трубе и присылает хозяину сообщение на телефон. Тут уже есть и управление (кран), и связь (сообщение).

Светофор с кнопкой. Над дорогой к школе горит зелёный для машин, пока пешеход не нажмёт кнопку на столбе. Сигнал от кнопки идёт по кабелю в серый шкаф у перекрёстка, там стоит контроллер. Он не переключает свет сразу: сначала проверяет, достаточно ли долго машины ехали. Потом – жёлтый, потом красный для машин и зелёный для людей. На больших перекрёстках под асфальтом или на столбах ещё стоят датчики машин: светофор видит, с какой стороны скопилась очередь, и даёт ей зелёный подольше.

Автопилот машины. Камеры смотрят на разметку. **Радар** – прибор, который посылает радиоволну и ловит её отражение, – меряет расстояние до машины впереди. Бортовой компьютер решает, сколько повернуть руль и нужно ли тормозить. Исполнители – электромотор рулевого управления, тормоза, двигатель. Тут счёт идёт на доли секунды: на 72 км/ч машина за каждую десятую долю секунды проезжает 2 метра.

Космический аппарат. Спутнику нужно, чтобы солнечные панели смотрели на Солнце. Датчик Солнца показывает, откуда идёт свет. Бортовой компьютер считает, на сколько повернуть. Исполнители – моторы, которые поворачивают панели, и **маховики** – тяжёлые колёса внутри спутника: раскручиваешь маховик в одну сторону, спутник поворачивается в другую.

С космосом есть своя трудность – расстояния. Радиосигнал летит со скоростью света, около 300 000 км в секунду. До Луны он добирается чуть больше чем за секунду. А до Марса – от 3 до 22 минут в одну сторону, смотря где планеты находятся на своих орбитах. Рулить марсоходом с Земли, как машинкой с пульта, не получится: пока картинка долетит, пока вернётся команда, он уже уедет. Поэтому марсоход сам объезжает камни, а с Земли ему присылают только цель: «доехать вон до той скалы». Чем дальше аппарат, тем больше решений должна принимать его собственная КФС.

Совет. Если в задаче спрашивают, КФС ли перед тобой, ищи все три роли. Компьютерная игра про гонки – «кибер» без «физики»: там ничего не происходит с настоящими вещами. Механические часы с кукушкой – «физика» без «кибера»: там нет ни датчика, ни программы.

1.6. Связь и управление – две сквозные технологии

НА ПАЛЬЦАХ

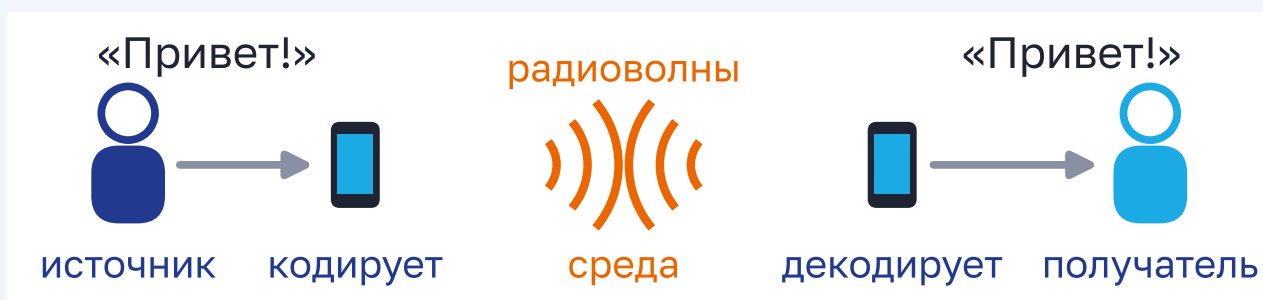


Рисунок 1.7 – Связь: сообщение превращается в сигнал, летит через среду и снова становится сообщением

Телефон превращает голос в радиоволны, второй телефон превращает их обратно в голос. Между ними – воздух и вышки.

Слово «кибернетика» прославил американский учёный Норберт Винер. В 1948 году вышла его книга «Кибернетика, или управление и связь в животном и машине». Название взято от греческого слова «кибернетес» – рулевой, тот, кто управляет кораблём. Две вещи из названия книги – **связь** и **управление** – проходят через всю сферу «Кибертех». Их называют **сквозными технологиями**: без них не работает ни одна КФС, какую ни возьми.

Связь – это как доставить сообщение от одного места к другому. Разработчики сферы раскладывают **канал связи** на пять частей:

- **источник** – тот, у кого есть сообщение (человек, датчик);
- **кодирующее устройство** – превращает сообщение в сигнал, который может лететь по среде (микрофон и телефон превращают голос в радиоволну);
- **физическая среда** – то, через что идёт сигнал: провод, воздух, вода, космос;

- **декодирующее устройство** – превращает сигнал обратно в сообщение;
- **получатель** – тот, кому сообщение нужно (контроллер, человек).

Среды бывают разные. По проводу идёт ток. По воздуху – радиоволны (Wi-Fi, Bluetooth, сотовая связь), звук и свет. Пульт от телевизора светит **инфракрасным** лучом – светом, который глаз не видит, а камера телефона видит: наведи на неё пульт и нажми кнопку. В каждой среде сигнал может ослабнуть или исказиться, и тогда сообщение дойдёт с ошибками. Как с этим бороться, расскажут главы 2–5.

Управление – это как заставить систему делать то, что нужно. Есть управление вслепую, по плану: фонарь по часам, стиральная машина крутит барабан 40 минут, что бы ни происходило. И есть управление с обратной связью, по показаниям датчиков: фонарь с датчиком света, котёл с термометром. Второе умнее, но и программу для него писать труднее: заранее нужно продумать, что делать в каждом случае. Как такие программы собирают в графической среде Кибериада IDE, узнаешь в главах 6 и 7.

Связь и управление всегда идут вместе. Датчик в теплице бесполезен, если его показания не дошли до контроллера. А команда «открыть клапан», потерянная по дороге, огурцы не польёт.

Запомни. Канал связи: источник → кодирующее устройство → среда → декодирующее устройство → получатель. Управление бывает по плану (вслепую) и с обратной связью (по датчикам). КФС нужно и то и другое: связь доставляет, управление решает.

1.7. Турнир юных киберфизиков

НА ПАЛЬЦАХ



Рисунок 1.8 – Турнир: одна команда передаёт сообщение звуком или инфракрасным лучом, другая принимает

На пути сигнала – шум и преграды. Выигрывает команда, чьё сообщение дошло без ошибок.

Турнир юных киберфизиков (ТЮК) – командное инженерное соревнование сферы. В программе кружка он встречается дважды: в первом модуле – чтобы сразу попробовать руками, и в двенадцатом, ближе к концу года, – чтобы проверить, чему научился. На турнире ко-

манды решают инженерную задачу, в которой надо наладить связь по разным каналам. Для этого есть комплекты «Акустика» (сообщение передают звуком) и «Инфракрасный сигнал» (передают невидимым светом), а ещё учебное устройство **КиберМишка**. На вебинаре сферы рассказывали, что у КиберМишки есть микрофоны, кнопки, акселерометр (датчик наклона и тряски, как в телефоне) и светодиодный экранчик, на котором он показывает, что услышал. Его программируют в Кибериаде IDE, а если настоящего Мишки нет – работают с его симулятором.

На финале НТО Юниоры основной тур проходит на компьютерах в симуляторах, а в регионах с оборудованием ещё устраивают «железный» хакатон: тот же канал связи, что настраивали в симуляторе, собирают из настоящих комплектов.

КиберМишку легко разложить на звенья из этой главы. Микрофон – датчик. Звук, который летит через комнату, – среда канала связи. Программа Мишки – обработка и управление. Экранчик – исполнитель: он зажигает буквы, и их видят люди.

Совет. На отборе про турнир вряд ли спросят, но его сюжеты встречаются в условиях задач. Встретишь незнакомое устройство – не пугайся названия, разложи его на датчик, передачу, обработку, управление и исполнитель.

1.8. Разбираем задачи

Задача 1.1 Автополив в теплице

разминка · звенья КФС, датчик и исполнитель

Бабушка уезжает с дачи на неделю, и внук собрал автополив для теплицы с огурцами. Как он работает, видно на рисунке. Каждой части системы (А–Д) поставь в соответствие её роль:

1 – датчик; 2 – передача; 3 – обработка; 4 – управление; 5 – исполнитель.

- А – электроклапан на трубе с водой: открывается и закрывается по команде;
- Б – радиомодуль: пересылает показания из теплицы в дом;
- В – датчик влажности, воткнутый в грядку;
- Г – прошивка в контроллере: сравнивает влажность почвы с порогом 30 %;
- Д – команда «открыть», которая идёт от контроллера по проводу к клапану.

В ответ введи номера ролей для частей А, Б, В, Г, Д подряд, без пробелов.

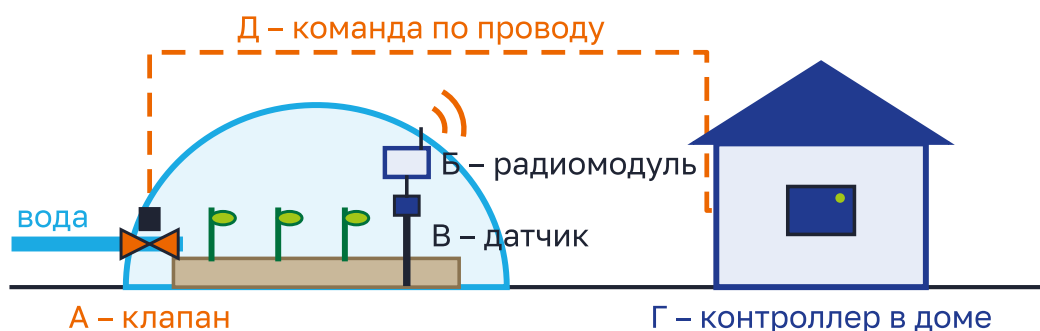


Рисунок 1.9 – Автополив: датчик в грядке, радиомодуль, контроллер в доме, клапан на трубе

Дано: пять частей системы А–Д и пять ролей 1–5.

Найти: строку из пяти цифр.

Решение

Шаг 1. Идём вслед за сигналом. Начинаем с места, где система узнаёт о мире. Сухость почвы замечает датчик влажности – это В, роль 1.

Шаг 2. Куда уходит показание. Датчик в теплице, контроллер в доме. Показание нужно доставить – это делает радиомодуль Б. Роль 2, передача.

Шаг 3. Кто думает. Число «влажность 22 %» само ничего не значит. Прошивка Г сравнивает его с порогом 30 % и решает: сухо, пора поливать. Это обработка, роль 3.

Шаг 4. Кто отдаёт приказ. Решение превращается в команду «открыть», которая бежит по проводу к клапану, – это Д. Управляющий сигнал, роль 4.

Шаг 5. Кто действует. Клапан А открывается, и вода течёт к огурцам. Он меняет мир – роль 5, исполнитель.

Шаг 6. Собираем ответ по порядку букв. А – 5, Б – 2, В – 1, Г – 3, Д – 4.

Себя легко проверить: пять частей, пять ролей, каждая цифра встретилась по одному разу. Если бы у тебя вышло два «датчика» и ни одной «передачи», стоило бы вернуться к шагам 1–2.

 **Типичная ошибка.** Записать цифры в порядке работы системы (В, Б, Г, Д, А → 1 2 3 4 5) вместо порядка букв. Условие просит ответ для частей А, Б, В, Г, Д – по алфавиту. Вторая ловушка – путать обработку и управление: «сравнить с порогом» – это ещё раздумья (3), а «команда клапану» – уже приказ (4).

Ответ: 52134.

Своя задача по программе сферы, модуль 1

Реши сам

1.1. Автоматические двери в супермаркете. Части: А – электромотор, который раздвигает створки; Б – датчик движения над входом; В – провод от датчика к плате управления; Г – программа на плате, которая решает «человек близко – открыть». Номера ролей те же, что в задаче 1.1: 1 – датчик; 2 – передача; 3 – обработка; 5 – исполнитель (управления отдельной частью здесь нет). Введи номера ролей для частей А, Б, В, Г подряд.

1.2. Робот-пылесос. Его части: 1 – бампер с кнопкой, которая нажимается при ударе о мебель; 2 – колёса с моторами; 3 – вращающаяся щётка с мотором; 4 – процессор с программой уборки; 5 – датчик на днище, который замечает край лестницы; 6 – модуль Wi-Fi, который управляет карту квартиры в телефон. а) Выпиши номера датчиков по возрастанию. б) Выпиши номера исполнителей по возрастанию.

Задача 1.2 Верно или нет

разминка · понятие КФС, роли частей

Выбери верные утверждения.

1. Фонарь, который каждый день включается в 17:00 по часам, сам замечает, что стемнело.
2. Датчик превращает то, что происходит в мире (свет, тепло, звук), в сигнал, понятный контроллеру.
3. Контроллер сам по себе, без исполнителя, может открыть воду в грядке.
4. Исполнитель – часть системы, которая меняет что-то в физическом мире: крутит колесо, открывает кран, зажигает лампу.
5. Если у системы отключить все датчики, она продолжит замечать, что происходит вокруг, только медленнее.
6. В КФС исполнитель меняет мир, а датчики замечают эти изменения, поэтому получается замкнутый круг.

В ответ введи номера верных утверждений по возрастанию, без пробелов.

Дано: шесть утверждений.

Найти: номера верных.

Решение

Шаг 1. Утверждение 1. Фонарь по часам не знает, светло на улице или темно, – у него нет датчика света. Он просто смотрит на часы. Неверно.

Шаг 2. Утверждение 2. Это определение датчика из раздела 1.1. Верно.

Шаг 3. Утверждение 3. Контроллер – «голова», у него нет «рук». Он может только отправить команду, а открывает воду клапан. Неверно.

Шаг 4. Утверждение 4. Определение исполнителя. Верно.

Шаг 5. Утверждение 5. Датчики – единственный способ системы узнать о мире. Отключишь их – система ослепнет совсем, и скорость тут ни при чём. Неверно.

Шаг 6. Утверждение 6. Это обратная связь из раздела 1.2: котёл нагрел комнату, термометр увидел. Верно.

Шаг 7. Ответ. Верные – 2, 4, 6.

 **Типичная ошибка.** Выбрать утверждение 1, потому что «фонарь же включается сам». Включаться сам умеет и будильник. Вопрос в том, **замечает** ли система, что происходит в мире. По часам – нет.

Ответ: 246.

Своя задача по программе сферы, модуль 1

Реши сам

1.3. Какие из систем – киберфизические? 1 – обычный выключатель света в комнате; 2 – котёл на даче, который по термометру держит в доме 20 °С; 3 – компьютерная игра, где ты ведёшь машинку по нарисованной трассе; 4 – шлагбаум, который открывается, когда камера узнала номер машины из списка жильцов; 5 – механические часы с кукушкой. Введи номера по возрастанию.

1.4. В умном доме бегают такие сигналы: 1 – датчик сообщает блоку «в комнате 17 °С»; 2 – блок посылает котлу «включить нагрев»; 3 – датчик протечки сообщает «на полу вода»; 4 – блок посылает крану на трубе «перекрыть воду»; 5 – датчик на двери сообщает «дверь открыта»; 6 – блок посылает лампе в прихожей «включиться». Выпиши номера управляющих сигналов по возрастанию.

Задача 1.3 Светофор по кнопке

цепочка КФС, порядок событий

На переходе у школы стоит светофор с кнопкой. Пока кнопку никто не нажал, для машин горит зелёный. Ниже – события, перепутанные местами. Расставь их в том порядке, в котором они происходят.

1. Для машин загорается жёлтый.
2. Пешеход нажимает кнопку на столбе.
3. Контроллер отправляет лампам команду переключиться.
4. По кабелю в шкаф управления приходит сигнал «кнопка нажата».
5. Для машин загорается красный, для пешеходов – зелёный.
6. Контроллер проверяет, что зелёный для машин горит уже не меньше 30 секунд.

В ответ введи номера событий в правильном порядке, без пробелов.

Дано: шесть событий 1–6.

Найти: их порядок.

Решение

Шаг 1. Найдём датчик. Всё начинается с того, что система узнаёт о пешеходе. Её «датчик» здесь – кнопка. Первое событие – 2.

Шаг 2. Передача. Кнопка на столбе, контроллер в шкафу. Сигнал идёт по кабелю – событие 4.

Шаг 3. Обработка. Контроллер не переключает свет сразу: сначала проверяет, достаточно ли ехали машины, – событие 6.

Шаг 4. Управление. Проверка пройдена, контроллер отдаёт приказ лампам – событие 3.

Шаг 5. Исполнители работают. Лампы переключаются. Сразу с зелёного на красный машинам нельзя – водитель не успеет затормозить. Поэтому сначала жёлтый (1), потом красный для машин и зелёный для пешеходов (5).

Шаг 6. Собираем. 2 → 4 → 6 → 3 → 1 → 5.

Звенья легли в порядке раздела 1.2: датчик, передача, обработка, управление, исполнитель. Если бы порядок вышел другим, например команда раньше сигнала от кнопки, стоило бы перепроверить.

 **Типичная ошибка.** Поставить событие 6 в самый конец или вовсе в начало: «контроллер следит за временем всегда». Следит – да, но проверку «можно ли уже переключать» он делает, когда узнал о нажатии. А ещё часто забывают про жёлтый и ставят 5 сразу после 3.

Ответ: 246315.

Своя задача по программе сферы, модуль 1

Реши сам

1.5. Фонарь с датчиком освещённости. Расставь события по порядку: 1 – лампа загорается; 2 – на улице темнеет; 3 – контроллер сравнивает освещённость с порогом и решает «пора»; 4 – датчик на крыше фиксирует, что света стало мало; 5 – контроллер подаёт команду на реле лампы (реле – выключатель, которым щёлкает электроника).

1.6. Робот-пылесос едет по комнате. Расставь события по порядку: 1 – колёса крутятся в разные стороны, робот разворачивается; 2 – бампер упирается в ножку стула, и кнопка в бампере нажимается; 3 – процессор получает сигнал «удар спереди»; 4 – датчик на днище видит, что пол снова под роботом; 5 – процессор решает «развернуться на 90°» и посылает команду моторам. Одно событие в этой истории лишнее – оно не может произойти, потому что робот не подъезжал к краю лестницы. Введи номера остальных четырёх событий в правильном порядке.

Задача 1.4 Пока автоматика думает

время реакции, скорость, перевод км/ч в м/с

В машине стоит система автоматического торможения. Радар опрашивается 20 раз в секунду. Передача показания в бортовой компьютер занимает 0,01 с, обработка – 0,04 с, а тормозам нужно 0,2 с, чтобы начать тормозить после команды. Машина едет со скоростью 54 км/ч. Сколько метров она проедет в худшем случае от появления препятствия до начала торможения? Ответ дай в метрах, округли до десятых.

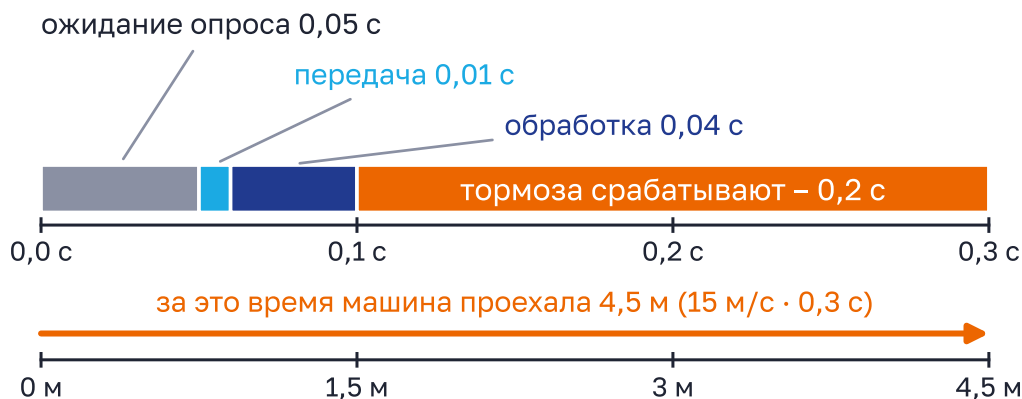


Рисунок 1.10 – Время реакции складывается из задержек. Всё это время машина едет с прежней скоростью

Дано: 20 опросов в секунду; $t_{\text{пер}} = 0,01 \text{ с}$; $t_{\text{обр}} = 0,04 \text{ с}$; $t_{\text{исп}} = 0,2 \text{ с}$; $v = 54 \text{ км/ч}$.

Найти: s – ?

Решение

Шаг 1. Период опроса. 20 раз в секунду – значит, между опросами $1/20 = 0,05 \text{ с}$.

Шаг 2. Худший случай. Препятствие появилось сразу после опроса. Радар заметит его только на следующем, через целый период: ожидание $0,05 \text{ с}$.

Шаг 3. Время реакции. Складываем все задержки: $t = 0,05 + 0,01 + 0,04 + 0,2 = 0,3 \text{ с}$.

Шаг 4. Скорость в м/с. Время у нас в секундах, значит, и скорость нужна в метрах в секунду: $v = 54/3,6 = 15 \text{ м/с}$.

Шаг 5. Путь. $s = v \cdot t = 15 \cdot 0,3 = 4,5 \text{ м}$.

Это примерно длина самой машины. Для сравнения: человеку от момента, когда он увидел препятствие, до нажатия на тормоз нужно около секунды – за это время машина проехала бы все 15 м. Автоматика выигрывает у водителя больше десяти метров.

⚠ **Типичная ошибка.** Подставить скорость в км/ч: $54 \cdot 0,3 = 16,2$ – и получить «16,2 м». Километры в час и секунды в одной формуле не живут. Ещё одна – забыть ожидание опроса и ответить $15 \cdot 0,25 = 3,75 \text{ м}$: это лучший случай, а спрашивали про худший.

Ответ: 4,5 м.

Своя задача по программе сферы, модуль 1

Реши сам

1.7. Школьный автобус едет по трассе со скоростью 72 км/ч . Система торможения реагирует на препятствие за $0,5 \text{ с}$. Сколько метров проедет автобус, прежде чем начнёт тормозить?

1.8. Робот-курьер едет по тротуару в Академгородке со скоростью 1,8 км/ч. Передача и обработка показаний его датчика занимают вместе 0,1 с, остановка моторов после команды – ещё 0,15 с. От появления пешехода до начала торможения робот должен проехать не больше 20 см, даже в худшем случае. Сколько раз в секунду как минимум нужно опрашивать датчик? Ответ – целое число.

Задача 1.5 Марсоход на пульте

задержка связи, скорость света, автономность

Инженер хочет вести марсоход «с пульта», как машинку. Сегодня между Землёй и Марсом 90 млн км. Радиосигнал летит со скоростью 300 000 км/с. Марсоход едет со скоростью 4 см/с. Он наехал на край ямы, камера сразу это сняла, и снимок полетел на Землю. Инженер увидел снимок и мгновенно нажал «стоп». Сколько метров проедет марсоход от момента съёмки до получения команды «стоп»?

Дано: $L = 90\,000\,000$ км; $c = 300\,000$ км/с; $v = 4$ см/с.

Найти: s – ?

Решение

Шаг 1. Сигнал в одну сторону. Время = путь / скорость: $t_1 = 90\,000\,000 / 300\,000 = 300$ с. Это 5 минут.

Шаг 2. Туда и обратно. Снимок летит на Землю 300 с, команда «стоп» летит на Марс ещё 300 с. Инженер реагирует мгновенно, так что всего $t = 300 + 300 = 600$ с – 10 минут.

Шаг 3. Путь марсохода. $s = 4 \cdot 600 = 2400$ см. Переводим в метры: $2400 / 100 = 24$ м.

Двадцать четыре метра – марсоход давно свалился бы в яму. А ведь 90 млн км – не самое большое расстояние до Марса: бывает и больше чем в четыре раза дальше. Поэтому настоящие марсоходы сами смотрят под колёса и сами объезжают опасности: с Земли им присылают только цель на день.

⚠ Типичная ошибка. Посчитать задержку только в одну сторону: $4 \cdot 300 = 1200$ см = 12 м. Но снимку нужно долететь до Земли, а команде – обратно на Марс, это два пути. И не забудь перевести сантиметры в метры: ответ «2400» без единиц – это не 2400 м.

Ответ: 24 м.

Своя задача по программе сферы, модуль 1

Реши сам

1.9. До Луны 384 000 км, радиосигнал летит со скоростью 300 000 км/с. Луноход-робот наехал на камень, камера сразу это сняла. Оператор на Земле реагирует мгновенно. Через сколько секунд после съёмки луноход получит ответную команду? Округли до сотых.

1.10. Космический аппарат отправил на Землю снимок. Ответ с Земли пришёл на аппарат через 50 минут после отправки снимка. Из них 6 минут инженеры на Земле изучали снимок и готовили команду. Радиосигнал летит со скоростью 300 000 км/с. На каком расстоянии от Земли аппарат? Ответ – в миллионах километров.

Задача 1.6* Дрон в тумане

датчики приближения, правило управления, выбор манёвра

Дрон-разведчик летит сквозь туман, камера ничего не видит. Спереди у него столбик из семи датчиков приближения, сверху вниз – датчики 1–7. Датчик 4 смотрит прямо по курсу, остальные – на ряды выше и ниже. Каждый датчик выдаёт 1, если в его ряду впереди препятствие, и 0, если ряд свободен.

Правило прошивки такое. Если свободен ряд 4 – лететь прямо (манёвр 0). Иначе искать ближайший свободный ряд: сначала на один ряд ниже, потом на один выше, потом на два ниже, на два выше, на три ниже, на три выше. Первый найденный – туда и сместиться: вверх на k рядов – манёвр $+k$, вниз – манёвр $-k$. Снижаться выгоднее, поэтому при равном расстоянии вниз проверяют раньше. После манёвра датчики снова отсчитывают ряды от нового курса.

Дрон получил пять показаний подряд (цифры датчиков 1–7 сверху вниз):

Таблица 1.2

Показание	Датчики 1–7
П1	0 1 1 1 1 1 1
П2	0 1 1 0 1 1 1
П3	1 0 1 1 1 1 0
П4	0 1 1 1 0 1 1
П5	1 0 1 1 1 0 1

На сколько рядов выше (+) или ниже (–) начального курса окажется дрон после пятого манёвра? Ответ – целое число со знаком.

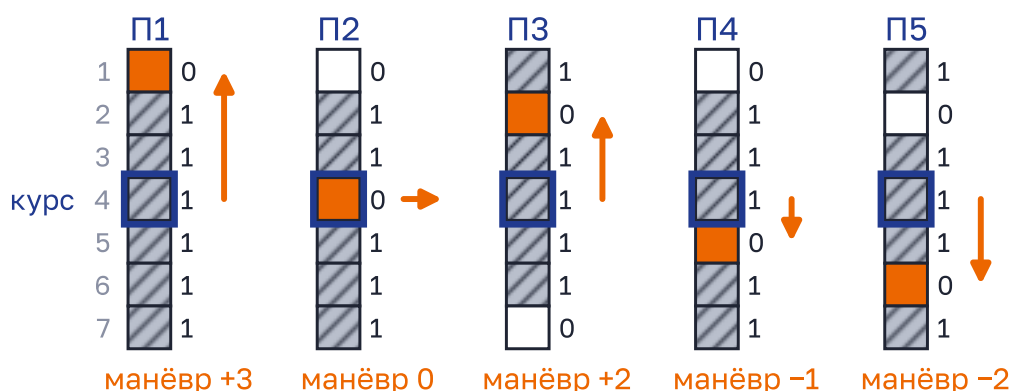


Рисунок 1.11 – Пять показаний датчиков. Оранжевым отмечен выбранный свободный ряд, стрелкой – манёвр

Дано: пять показаний, правило поиска «ниже – выше, от ближних к дальним».

Найти: итоговое смещение в рядах.

Решение

Шаг 1. Порядок проверки. Выпишем, в каком порядке прошивка смотрит на датчики: 4, потом 5 (на ряд ниже), 3 (на ряд выше), 6, 2, 7, 1. Первый ноль в этом порядке и решает манёвр.

Шаг 2. П1: 0 1 1 1 1 1 1. Датчик 4 – единица, 5 – единица, 3 – единица, 6 – единица, 2 – единица, 7 – единица, 1 – ноль. Датчик 1 на три ряда выше курса: манёвр +3.

Шаг 3. П2: 0 1 1 0 1 1 1. Датчик 4 показывает 0 – курс свободен, летим прямо: манёвр 0. То, что свободен и ряд 1, значения не имеет.

Шаг 4. П3: 1 0 1 1 1 1 0. 4 – занят, 5 – занят, 3 – занят, 6 – занят, 2 – свободен. Датчик 2 на два ряда выше: манёвр +2. Свободен и ряд 7, но он на три ряда ниже – дальше, до него очередь не дошла.

Шаг 5. П4: 0 1 1 1 0 1 1. 4 – занят, 5 – свободен. Манёвр –1.

Шаг 6. П5: 1 0 1 1 1 0 1. 4, 5, 3 заняты. На расстоянии два ряда свободны оба: 6 (ниже) и 2 (выше). По правилу вниз проверяют раньше – выбираем 6. Манёвр –2.

Шаг 7. Складываем. Каждое показание отсчитано от нового курса, поэтому манёвры просто складываются: $+3 + 0 + 2 - 1 - 2 = +2$. По пути дрон побывал на высотах +3, +3, +5, +4, +2 ряда от начального курса.

⚠ Типичная ошибка. В П5 выбрать ряд 2, потому что он «выше в списке», и получить манёвр +2 – тогда итог +6. При равном расстоянии правило велит снижаться. Вторая ошибка – в П3 взять ряд 7, «ведь снижаться выгоднее»: но выгода работает только при равном расстоянии, а ряд 2 ближе.

Ответ: 2.

По мотивам задачи первого отборочного этапа НТО 2025/26, профиль «Технологии беспроводной связи», «Крутое пике»

Реши сам

1.11. Дрон из задачи 1.6 получил показание 0 1 1 1 1 1 0. Какой манёвр он выполнит? Ответ – целое число со знаком.

1.12.* Дрон из задачи 1.6 с тем же правилом получил пять показаний: 1 1 1 1 0 1 1; 1 0 1 1 1 1 1; 1 1 0 1 1 1; 0 1 1 1 1 1 0; 1 1 0 1 0 1 1. Подъём на один ряд тратит 3 единицы заряда, спуск на один ряд – 1 единицу, полёт прямо – ничего. Сколько единиц заряда потратит дрон на все пять манёвров?

Задача 1.7* Обогреватель в теплице

два порога, память состояния, обратная связь

Весной ночью в теплице на даче может ударить заморозок. Обогреватель управляется так: если температура **ниже** 12 °С – включить; если **выше** 16 °С – выключить; иначе – оставить как есть. Контроллер смотрит на термометр раз в 10 минут и до следующего замера ничего не меняет. В 20:00 обогреватель выключен. Замеры с 20:00 до 21:50:

Таблица 1.3

Время	°С	Время	°С
20:00	15	21:00	17
20:10	13	21:10	15
20:20	11	21:20	13
20:30	12	21:30	11
20:40	14	21:40	10
20:50	16	21:50	12

Сколько минут работал обогреватель с 20:00 до 22:00?

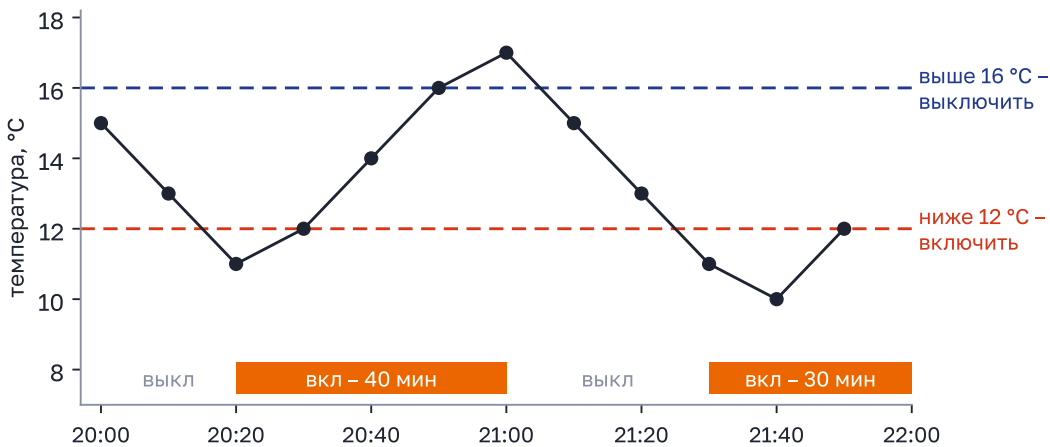


Рисунок 1.12 – Температура в теплице и работа обогревателя. Между порогами 12 и 16 °С он не меняет состояние

Дано: пороги 12 °С (включить) и 16 °С (выключить); 12 замеров через 10 мин; в начале выключен.
Найти: время работы, мин.

Решение

Шаг 1. Правило на каждый замер. Это два порога из раздела 1.3. Меньше 12 – «вкл». Больше 16 – «выкл». От 12 до 16 включительно – оставить то, что было. Каждый замер решает судьбу следующих 10 минут.

Шаг 2. Идём по замерам.

- 20:00 – 15 °С: между порогами, остаётся как было – выкл.
- 20:10 – 13 °С: между порогами – выкл.
- 20:20 – 11 °С: меньше 12 – **вкл**.
- 20:30 – 12 °С: не меньше 12 и не больше 16 – остаётся вкл.
- 20:40 – 14 °С: остаётся вкл.

- 20:50 – 16 °C: не больше 16 – остаётся вкл.
- 21:00 – 17 °C: больше 16 – **выкл.**
- 21:10 – 15 °C, 21:20 – 13 °C: между порогами – выкл.
- 21:30 – 11 °C: **вкл.**
- 21:40 – 10 °C: вкл. 21:50 – 12 °C: остаётся вкл.

Шаг 3. Считаем. Обогреватель был включён после замеров 20:20, 20:30, 20:40, 20:50 – это 4 отрезка, и после замеров 21:30, 21:40, 21:50 – ещё 3. Всего $4 + 3 = 7$ отрезков по 10 минут: $7 \cdot 10 = 70$ мин.

Проверка кодом. Это первая программа в книге, разберём её построчно:

```
temps = [15, 13, 11, 12, 14, 16, 17, 15, 13, 11, 10, 12]
heater = False           # в начале выключен
minutes = 0
for t in temps:          # берём замеры по одному
    if t < 12:
        heater = True    # холодно – включить
    elif t > 16:
        heater = False   # жарко – выключить
    # иначе ничего не делаем: heater помнит прежнее
    if heater:
        minutes += 10    # ещё 10 минут работы
print(minutes)           # 70
```

Строка 1 – список замеров. Строка 2 – **переменная** `heater` («обогреватель»): ячейка памяти, в которой лежит `True` (включён) или `False` (выключен). Строка 3 – счётчик минут. Строка 4 – цикл: для каждого замера `t` по очереди выполняются строки с отступом. `if` значит «если», `elif` – «а иначе если». Строки, начинающиеся с `#`, – пояснения, Python их пропускает. Главное здесь то, чего в программе нет: для температуры от 12 до 16 нет никакой команды, и `heater` сохраняет старое значение. Так система и помнит своё состояние.

Проверим на здравый смысл: из 120 минут обогреватель проработал 70 – больше половины. Холодная весенняя ночь, температура дважды падала ниже 12 °C – похоже.

⚠ Типичная ошибка. Забыть про память и считать, что обогреватель работает только тогда, когда ниже 12 °C. Таких замеров три (11, 11, 10) – выходит 30 мин. Между порогами обогреватель продолжает делать то, что делал. Вторая ловушка – замеры 12 и 16 °C, которые попали точно на пороги: «ниже 12» и «выше 16» означает строго меньше и строго больше, поэтому на них ничего не переключается.

Ответ: 70 мин.

Своя задача по программе сферы, модуль 1

Реши сам

1.13. Обогреватель включается при температуре ниже 10°C и выключается при температуре выше 14°C , между порогами состояние не меняется. Замеры идут раз в 15 минут, в начале обогреватель выключен. Температуры: 12, 9, 10, 13, 14, 15, 11, 9, 8, 12°C . Сколько минут проработал обогреватель за эти 150 минут?

1.14.* Холодильник на даче устроен наоборот: компрессор (он охлаждает) включается, если внутри **выше** 6°C , и выключается, если **ниже** 3°C ; между порогами ничего не меняется. Замеры раз в 5 минут, в начале компрессор выключен. Температуры: 5, 7, 6, 4, 3, 2, 4, 6, 7, 5°C . Сколько минут работал компрессор за эти 50 минут?

Сигнал и его параметры



Звонок с урока, мигание пульта, писк микроволновки – всё это сигналы. Здесь ты научишься читать их графики: как сильно сигнал качается, как часто повторяется, сколько раз в секунду его измеряет прибор и когда он перескочил через порог. На отборе такие задачи приходят графиком или рядом чисел, а в ответ просят одно-два числа: частоту в герцах, число срабатываний, время в миллисекундах.

2.1. Явление и сигнал

НА ПАЛЬЦАХ

явление



лампа горит весь вечер

сигнал



→ время

три вспышки = «ужин готов!»

Рисунок 2.1 – Окно дачного домика вечером. Слева лампа просто горит, справа три вспышки

Свет один и тот же. Но про три вспышки вы с мамой договорились заранее – и свет превратился в сообщение.

Гром, ветер, свет лампы, тепло от батареи – это **физические явления**. Они просто происходят. Явление становится **сигналом**, когда в него вкладывают смысл и передают кому-то: договорились, что три вспышки фонарика значат «ужин готов», – и мигание стало сообщением. Школьный звонок – тоже сигнал: звук сам по себе ничего не значит, но все знают, что после него начинается перемена.

Сигнал – это **изменение физической величины, которое несёт информацию**. Величина может быть любой: громкость звука, яркость света, напряжение в проводе, температура, – но она должна меняться. Термометр, который неделю показывает одни и те же 20 градусов, ничего нового не расскажет. А если за ночь столбик упал до минус 30, это уже новость: пора доставать пуховик.

У любого сигнала есть три участника:

- **источник** – кто посылает: человек с фонариком, пульт от телевизора, динамик;
- **среда** – через что сигнал идёт: воздух, вода, провод, пустое пространство;
- **приёмник** – кто ловит: глаз, ухо, микрофон, датчик.

Сигнал может рассказывать и о самой среде, через которую прошёл. Эхолот на катере на Обском море посылает вниз короткий звук и слушает эхо от дна. Чем дальше эхо шло обратно, тем глубже в этом месте. Эхо вернулось слабым и «размазанным» – дно илистое или под катером косяк рыбы.

Электроника не слышит звук и не видит свет. Она понимает только электрический сигнал – изменение **напряжения**. Напряжение – это «напор», с которым электричество толкается по проводу; его меряют в вольтах (В). Поэтому в каждом устройстве стоит **датчик** – прибор, который переводит свет, звук или тепло в напряжение. У датчика освещённости (фоторезистора) от света меняется сопротивление – то, насколько сильно он мешает току течь, – и напряжение на нём растёт или падает. В микрофоне звук качает тонкую пластинку, и в проводе получается дрожащее напряжение, похожее на сам звук. У КиберМишки, устройства сферы, есть микрофоны, ИК-датчик (ловит инфракрасный свет, как телевизор ловит сигнал пульта) и акселерометр (чувствует наклон и тряску).

Запомни. Сигнал – изменение величины, в которое вложили смысл. Путь сигнала: источник → среда → приёмник. Контроллер понимает только напряжение, поэтому датчик переводит в него свет, звук и тепло.

2.2. Аналоговый и цифровой сигнал

НА ПАЛЬЦАХ

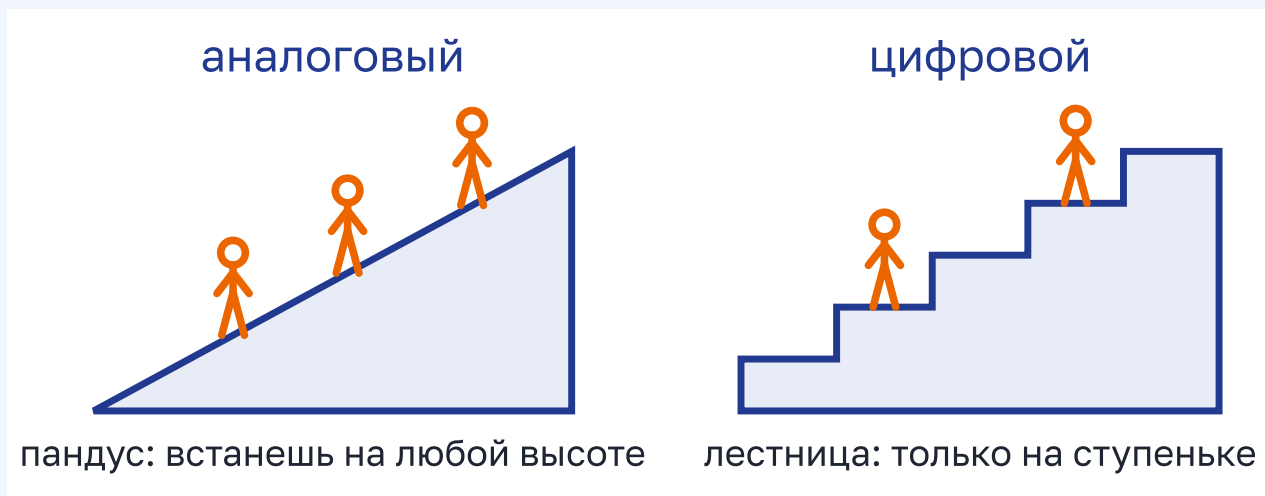


Рисунок 2.2 – Пандус и лестница

На пандусе в магазине можно остановиться на любой высоте. На лестнице – только на ступеньке, между ними не встанешь. Аналоговый сигнал – пандус, цифровой – лестница.

Аналоговый сигнал меняется плавно и может принять любое значение. Твой голос, температура за окном, стрелка на старых весах, столбик в стеклянном градуснике – всё это аналоговое. Между 36,6 и 36,7 градуса столбик пройдёт через 36,61, 36,62 и через все значения, какие только можно придумать.

Цифровой сигнал – это набор отдельных чисел. Промежуточных значений у него нет, как нет полступеньки на лестнице. Электронный градусник показывает 36,6 или 36,7, а 36,63 показать не может: его «ступенька» – одна десятая градуса. Самый простой цифровой сигнал – из двух уровней, 0 и 1: кнопка нажата или нет, свет есть или нет.

Мир вокруг нас аналоговый, а компьютер работает только с числами. Поэтому прибор превращает аналоговый сигнал в цифровой в два хода. Сначала **измеряет** его через равные промежутки времени – об этом раздел 2.4. Потом **округляет** каждое измерение до ближайшей своей ступеньки. Датчик может, например, выдавать целые числа от 0 до 1023: тогда напряжение 2,4837 В превратится в какое-то одно целое число, а мелкие отличия пропадут.

Зачем тогда вообще цифра, если она теряет подробности? Числа легко хранить, копировать и передавать без искажений. Помеха чуть-чуть изменила напряжение – аналоговый звук захрипит. А цифровой приёмник всё равно узнает, 0 пришёл или 1, если помеха меньше расстояния между ступеньками.

Запомни. Аналоговый – плавный, любые значения (пандус). Цифровой – отдельные числа, между ними ничего нет (лестница). Чтобы получить из аналогового цифровой, сигнал измеряют через равные промежутки и округляют до ступеньки.

2.3. Колебания: амплитуда, период, частота, фаза

НА ПАЛЬЦАХ



туда и обратно
= один период

сколько раз за секунду
= частота

Рисунок 2.3 – Качели в трёх положениях: в середине и в двух крайних точках

Качели отходят от середины на одно и то же расстояние – это амплитуда. Туда и обратно – один период. А сколько таких «туда-обратно» уложится в секунду – частота.

Многие сигналы повторяются: качели, струна гитары, звук, ток в розетке. Такое повторяющееся движение называют **колебанием**. Самое простое и плавное колебание рисуется волнистой линией – **синусоидой**. Её ещё называют **гармоническим сигналом**. Почему она так важна, узнаешь в главе 3: из синусоид можно сложить любой сложный звук.

Сигнал рисуют **графиком**. По горизонтали – время, по вертикали – значение сигнала в этот момент (напряжение, громкость, число с датчика). Посмотри на рисунок: на нём подписано всё, что нужно уметь находить.

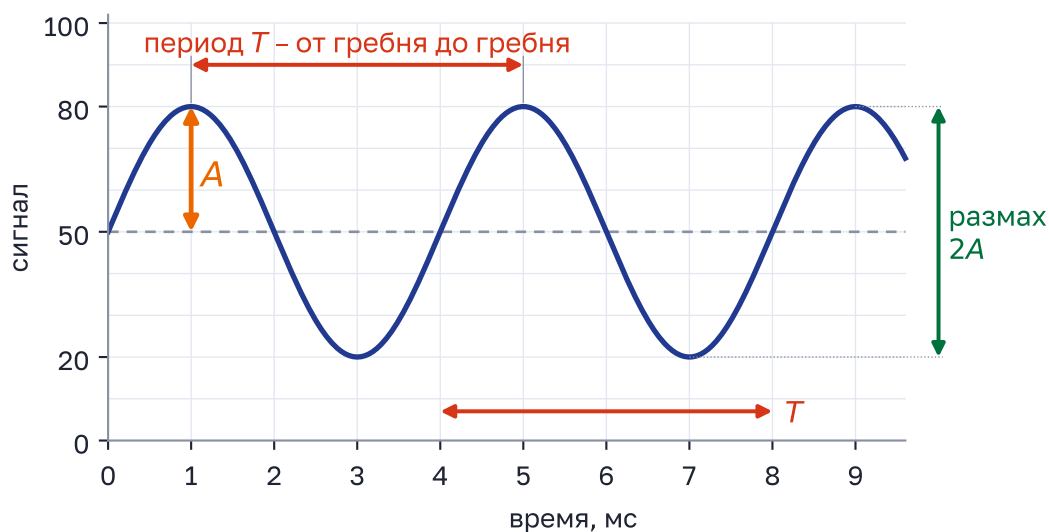


Рисунок 2.4 – Синусоида и её параметры. Пунктир – средний уровень, вокруг которого качается сигнал

Средний уровень – линия, вокруг которой сигнал качается. На рисунке это 50. Он совсем не обязан быть нулём: у датчика, который выдаёт числа от 0 до 1023, тишина часто записывается числом где-то посередине.

Амплитуда A – на сколько сигнал отходит от среднего уровня в самой высокой точке (её называют **гребнем**). На рисунке сигнал поднимается до 80, средний уровень 50, значит, $A = 80 - 50 = 30$. Расстояние от самой низкой точки до самой высокой – **размах** – вдвое больше: $80 - 20 = 60 = 2A$. Если известны только наибольшее и наименьшее значения, то

$$A = \frac{\text{наибольшее} - \text{наименьшее}}{2}, \quad \text{средний уровень} = \frac{\text{наибольшее} + \text{наименьшее}}{2}.$$

Период T – время одного полного колебания. Мерить его можно от гребня до следующего гребня, от впадины до впадины или от любой точки до следующей такой же, где сигнал снова идёт в ту же сторону. На рисунке гребни в 1 мс и 5 мс, значит, $T = 5 - 1 = 4$ мс. А ещё по нижней стрелке: сигнал поднимается через средний уровень в 4 мс и снова в 8 мс, те же 4 мс.

Частота f – сколько полных колебаний проходит за одну секунду. Единица частоты – **герц** (Гц), по фамилии учёного Генриха Герца: 1 Гц – это одно колебание в секунду. Частота и период связаны простым правилом: чем короче одно колебание, тем больше их влезает в секунду. Словами – частота равна единице, делённой на период:

$$f = \frac{1}{T}, \quad T = \frac{1}{f},$$

где T – в секундах, а f – в герцах. Если период дан в миллисекундах, **сначала переведи его в секунды**: 1 мс = 0,001 с. На рисунке $T = 4$ мс = 0,004 с, и $f = 1 : 0,004 = 250$ Гц.

Ещё два пересчёта. 1 кГц (килогерц) = 1000 Гц, 1 МГц (мегагерц) = 1 000 000 Гц. Сколько колебаний N пройдёт за время t : $N = f \cdot t$. Ток в розетке меняется с частотой 50 Гц, его период $1 : 50 = 0,02$ с = 20 мс, и за минуту он успеет качнуться $50 \cdot 60 = 3000$ раз.

Что меняется, если меняются амплитуда и частота:

- у **звука** амплитуда – это громкость, частота – высота тона. Писк комара – высокая частота, гудение трансформаторной будки – низкая. Человек слышит примерно от 20 Гц до 20 000 Гц, а нота «ля» первой октавы, по которой настраивают инструменты, – это 440 Гц;
- у **света** амплитуда – яркость, частота – цвет.



Рисунок 2.5 – Серым – один и тот же исходный сигнал. Синим – что будет, если увеличить амплитуду, увеличить частоту или сдвинуть фазу

качели одинаковые, а движения сдвинуты на четверть периода

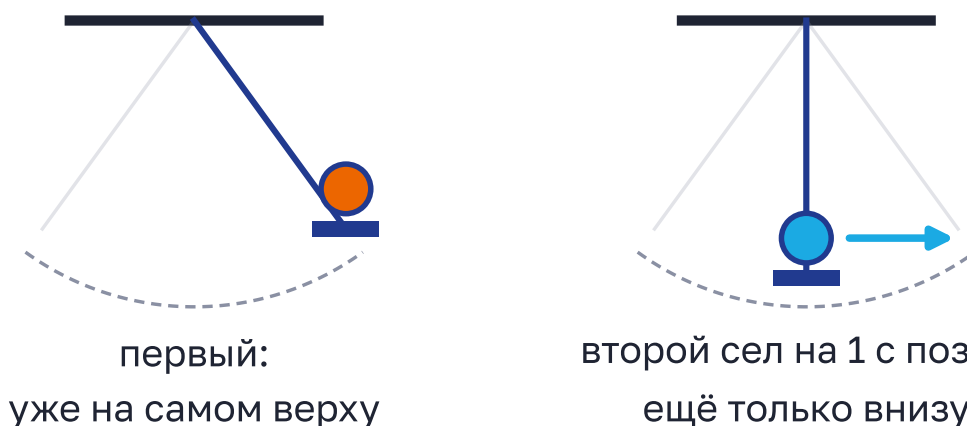


Рисунок 2.6 – Двое на одинаковых качелях. Первый уже на самом верху, второй сел на секунду позже и только проносится вниз

Качели одинаковые, размах одинаковый. Отличается только одно: кто на сколько опоздал. Это опоздание и есть сдвиг по фазе.

Фаза. Вернись к качелям на картинке. Амплитуда и период у ребят одни и те же, но когда первый взлетел на самый верх, второй только проносится через середину. Их колебания **сдвинуты по фазе**. Фаза говорит, в какой части своего колебания сигнал находится сейчас: на подъёме, на гребне, на спуске. Сдвиг удобно мерить долей периода. Пусть качели делают одно «туда-обратно» за 4 с – тогда секунда опоздания составляет четверть периода: $1:4 = 0,25$. Второй доберётся до верха, когда первый уже вернётся к середине.

Самый интересный сдвиг – на **половину периода**, его называют **противофазой**. Один сигнал вверх, когда другой вниз, и наоборот. Сложи два таких сигнала с одинаковой амплитудой – и они погасят друг друга. На этом работают наушники с шумоподавлением: микрофон слушает шум вокруг, наушник играет тот же звук, сдвинутый на полпериода, и шум в ухе почти пропадает.

Для любопытных. Синусоиду записывают формулой $s(t) = A \sin(2\pi f t + \varphi)$. Здесь A – амплитуда, f – частота, t – время, φ (греческая «фи») – начальная фаза: с какой точки колебания сигнал начался. В курсе частоту обозначают буквой ν («ню» вместо f) и пишут $\omega = 2\pi \nu$ – это циклическая частота; смысл тот же. Считают формулу в Excel: в одной ячейке `=E$1*SIN(2*ПИ()*E$2*A2+E$3)`, где амплитуда, частота в герцах и начальная фаза лежат в ячейках E1–E3, а время в секундах – в столбце A. Фазу ставят **в радианах**: полкруга – 3,1416 (π), четверть круга – 1,5708 ($\pi/2$). Ответ потом округляют так, как сказано в задаче: вышло 29,999999 – пиши 30.

Как читать график сигнала. Порядок всегда один.

1. Прочитай подписи осей. В чём время: в секундах, миллисекундах? Сколько стоит одна клеточка?

2. Найди наибольшее и наименьшее значения. Из них – средний уровень и амплитуду.
3. Найди два гребня **подальше друг от друга** и посчитай, сколько полных колебаний между ними. Раздели время между гребнями на это число – получишь период. Так точнее, чем мерить одно колебание: ошибка в полклеточки делится на число колебаний.
4. Переведи период в секунды и найди частоту: $f = 1/T$.

Запомни. Амплитуда – от среднего уровня до гребня, половина размаха. Период – время одного колебания, частота – сколько колебаний за секунду: $f = 1/T$, и период перед этим переводят в секунды. Громкость – амплитуда, высота тона – частота. Сдвиг на половину периода – противофаза: сигналы гасят друг друга.

2.4. Дискретизация: сигнал по точкам

НА ПАЛЬЦАХ

мячик на резинке: снимок через равные промежутки

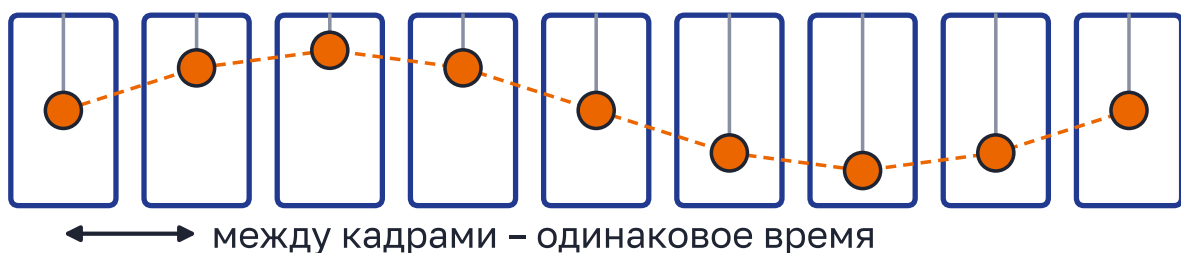


Рисунок 2.7 – Мячик на резинке, снятый несколько раз через одинаковое время

Кино – это много фотографий подряд. Компьютер тоже «фотографирует» сигнал через равные промежутки и хранит только снимки. Соединишь их – получишь движение.

Компьютер не может следить за сигналом непрерывно. Он измеряет его время от времени и записывает числа. Одно такое измерение называется **отсчётом**, а сам способ – измерять сигнал через равные промежутки времени – **дискретизацией** (от латинского слова «прерывистый»).

Время между соседними отсчётами называют **шагом** и обозначают Δt («дельта тэ»). А сколько отсчётов прибор делает за одну секунду – это **частота дискретизации** f_d , её ещё называют частотой отсчётов. Единица у неё та же – герц. Шаг и частота дискретизации связаны так же, как период и частота: $f_d = 1/\Delta t$. Термометр на метеостанции, который меряет температуру раз в 10 минут, делает 6 отсчётов в час и $6 \cdot 24 = 144$ отсчёта в сутки. Музыку на компакт-диске записывали с частотой дискретизации 44 100 Гц: 44 100 отсчётов в каждую секунду для каждого из двух каналов, левого и правого.

Две формулы пригодятся в задачах. Сколько отсчётов наберётся за время t :

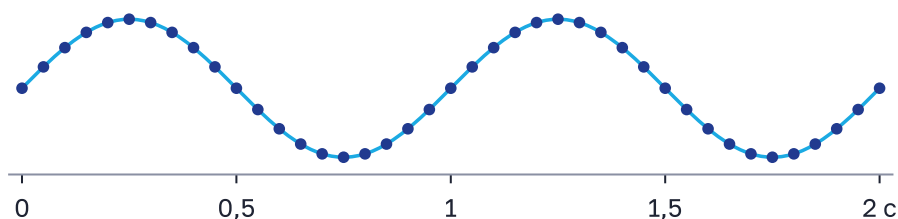
$$N = f_d \cdot t.$$

Сколько отсчётов придётся на одно колебание сигнала с частотой f :

$$\text{отсчётов на период} = \frac{f_d}{f}.$$

Зачем считать отсчёты на период? По ним видно, хватит ли их, чтобы разглядеть сигнал.

1 колебание в секунду, 20 отсчётов в секунду: по точкам форма видна



9 колебаний в секунду, а отсчётов только 10: точки рисуют чужую медленную волну

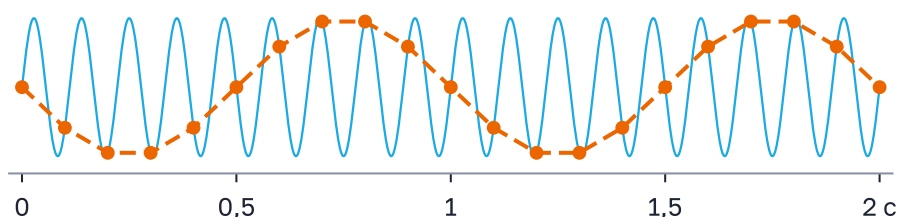


Рисунок 2.8 – Сверху: отсчётов много, и точки ложатся на волну. Снизу: отсчётов меньше, чем колебаний, и соединённые точки дают совсем другую, медленную волну

Сверху на одно колебание приходится 20 отсчётов, и точки повторяют форму волны. Снизу сигнал делает 9 колебаний в секунду, а прибор измеряет только 10 раз в секунду. Между двумя отсчётами волна успевает пройти почти целое колебание, и каждая следующая точка попадает в неё чуть в другом месте. Соединишь точки – выйдет медленная волна, одно колебание в секунду. Такого сигнала не было, его нарисовали редкие точки. По той же причине в кино колёса быстрой машины иногда будто крутятся медленно или даже назад.

Правило такое: **отсчётов на один период должно быть больше двух**. Иначе быстрый сигнал «притворится» медленным. Поэтому для компакт-диска и выбрали 44 100 Гц: человек слышит звуки до 20 000 Гц, и на каждое колебание самого высокого из них приходится больше двух отсчётов ($44\,100 : 20\,000 = 2,2$). А чтобы по точкам ещё и разглядеть форму волны, берут 10–20 отсчётов на период.

Запомни. Отсчёт – одно измерение, дискретизация – измерения через равные промежутки. Частота дискретизации – сколько отсчётов за секунду: $N = f_d \cdot t$. На период сигнала приходится f_d / f отсчётов, и их должно быть больше двух.

2.5. Детектирование: порог, период, шум

НА ПАЛЬЦАХ

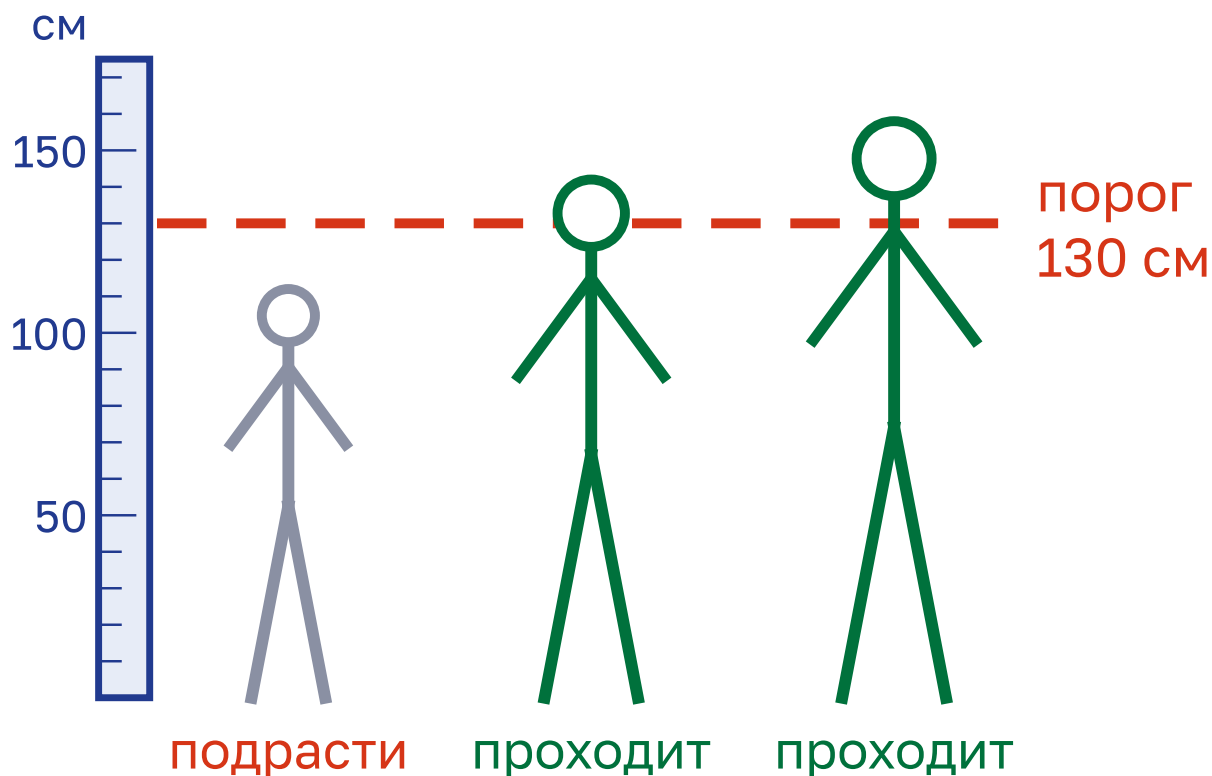


Рисунок 2.9 – Атракцион «рост от 130 см»

Линия на стене – это порог. Макушка выше линии – проходишь, ниже – подожди годик. Датчик решает так же, только вместо роста у него значение сигнала.

Приёмнику мало поймать сигнал, из него нужно достать то, ради чего сигнал посылали. Это называется **детектированием**: выделить из сигнала полезное – «есть сигнал или нет», «какой высоты звук». Детектируют по уровню или по периоду.

Детектирование по уровню (его ещё называют амплитудным). Выбирают число – **порог срабатывания**. Сигнал выше порога – устройство считает, что «есть» (1), ниже – «нет» (0). Так фонарь во дворе решает, стемнело ли, а автоматическая дверь в супермаркете – подошёл ли кто-то. После сравнения с порогом аналоговый сигнал превращается в цифровой из нулей и единиц.

Сколько раз сработал датчик? Считать нужно не точки выше порога, а **переходы снизу вверх**: предыдущий отсчёт был ниже порога, а следующий – уже не ниже. Громкий хлопок может длиться три отсчёта подряд, но это один хлопок. Если отсчёт равен порогу, смотри условие: там скажут, считать его «выше» или «ниже».

В какой момент это случилось? Если первый отсчёт сделан в момент 0, удобно нумеровать отсчёты с нуля: номер 0 – в самом начале, номер 1 – через один шаг. Тогда **время отсчёта = номер · шаг**. Отсчёт номер 3 при шаге 50 мс сделан в момент $3 \cdot 50 = 150$ мс, а не 200 мс.

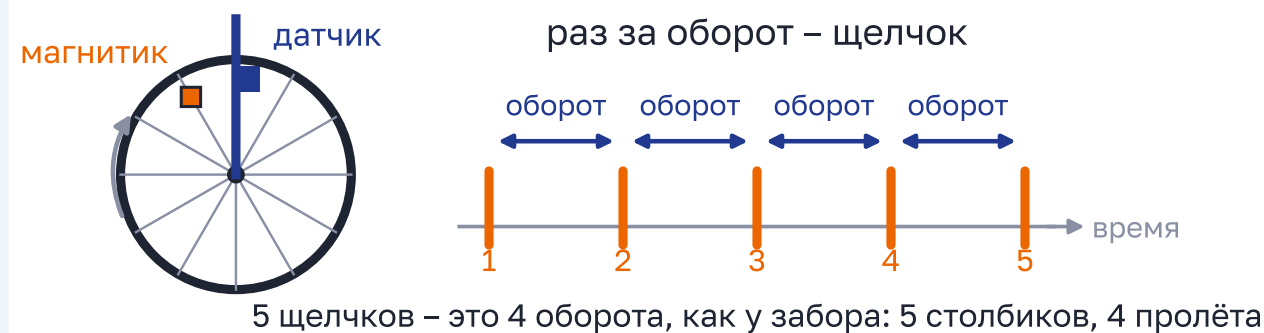


Рисунок 2.10 – Колесо с магнитиком и лента времени: пять щелчков, между ними четыре оборота

Магнитик пролетает мимо датчика раз за оборот – щёлк. Считать надо промежутки между щелчками, а не сами щелчки.

Детектирование по периоду (частотное). Здесь неважно, насколько сигнал сильный, – важно, как часто он повторяется. Прибор меряет время между одинаковыми событиями и находит период, а по нему – частоту. Так устроен велокомпьютер: на спице колеса магнитик, на вилке датчик. Время между щелчками – период оборота. Зная окружность колеса, прибор считает скорость. Похожие датчики стоят на колёсах автомобиля в системе АБС: по щелчкам она замечает, что колесо при торможении вдруг перестало крутиться, и чуть отпускает тормоз.

Щелчки и промежутки путают постоянно. У забора из 5 столбиков 4 пролёта – и пять щелчков дают **четыре** периода.

Шум. Настоящий сигнал с датчика никогда не бывает гладким. К нему примешивается **шум** (**помеха**) – всё лишнее, что попало по дороге: дрожание руки, соседний прибор, мелкие скачки внутри самого датчика. Вдали от порога шум не мешает. А вблизи порога сигнал начинает прыгать через него туда-сюда, и датчик срабатывает много раз вместо одного.

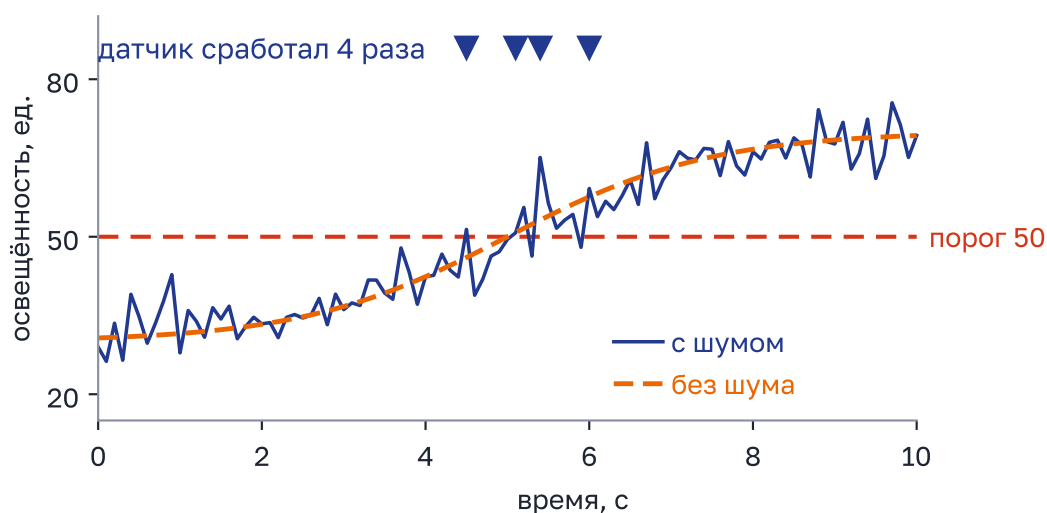


Рисунок 2.11 – Утро: освещённость медленно растёт и один раз переходит порог. Но из-за шума датчик сработал четыре раза

Что с этим делают:

- ставят **два порога**: включить выше одного, выключить ниже другого (так было с обогревателем в главе 1);
- требуют **подтверждения**: засчитывают срабатывание, только если сигнал продержался выше порога несколько отсчётов подряд. Короткий одиночный скачок так не пройдёт;
- **сглаживают** сигнал, прежде чем сравнивать его с порогом. Как это делает фильтр низких частот – в главе 3.

Запомни. По уровню – сравниваем с порогом, по периоду – меряем время между повторами. Срабатывание – переход снизу вверх, а не каждая точка выше порога. Время отсчёта = номер · шаг, нумерация с нуля. n щелчков – это $n - 1$ промежуток.

Совет. Если в задаче ряд чисел и порог, выпиши под каждым числом 0 или 1 (ниже или не ниже порога). Потом просто найди все места, где 0 сменяется на 1. Так не пропустишь ни одного перехода.

2.6. Разбираем задачи

Задача 2.1 Сколько раз в секунду

разминка · период и частота, перевод мс в с

Катя включила на компьютере генератор тона – программу, которая играет чистый звук и рисует его график. По графику видно, что одно колебание длится 2 мс. Какая частота у этого звука? Ответ дай в герцах.

Дано: $T = 2$ мс.

Найти: f – ?

Решение

Шаг 1. Период в секунды. В одной секунде 1000 мс, поэтому $T = 2 : 1000 = 0,002$ с.

Шаг 2. Частота. $f = 1/T = 1 : 0,002 = 500$ Гц.

То же можно понять и без деления на дробь. Одно колебание – 2 мс, а в секунде 1000 мс. Сколько раз 2 мс уложатся в 1000 мс? $1000 : 2 = 500$ раз. Значит, 500 колебаний в секунду.

Звук 500 Гц человек слышит хорошо: это чуть выше ноты «ля» (440 Гц), примерно середина клавиатуры пианино.

⚠ Типичная ошибка. Поделить единицу на 2 и ответить «0,5 Гц». Миллисекунды не перевели в секунды, и ответ вышел в 1000 раз меньше. Звук в полгерца человек не услышит: это одно колебание за две секунды, так медленно качаются качели, а не пищалка.

Ответ: 500 Гц.

Своя задача по программе сферы, модуль 2

Реши сам

2.1. Ток в розетке меняется с частотой 50 Гц. Сколько миллисекунд длится один период?

2.2. Два динамика играют один и тот же тон 250 Гц, но второй начал на 2 мс позже первого. Какую часть периода составляет это отставание? Ответ – десятичная дробь. Что услышит человек посередине между динамиками, если амплитуды звуков у него одинаковые?

Задача 2.2 Период по графику

чтение графика, амплитуда, период и частота

На рисунке – сигнал с микрофона, записанный, когда рядом звучала струна. а) Найди амплитуду сигнала. б) Найди частоту звука в герцах.

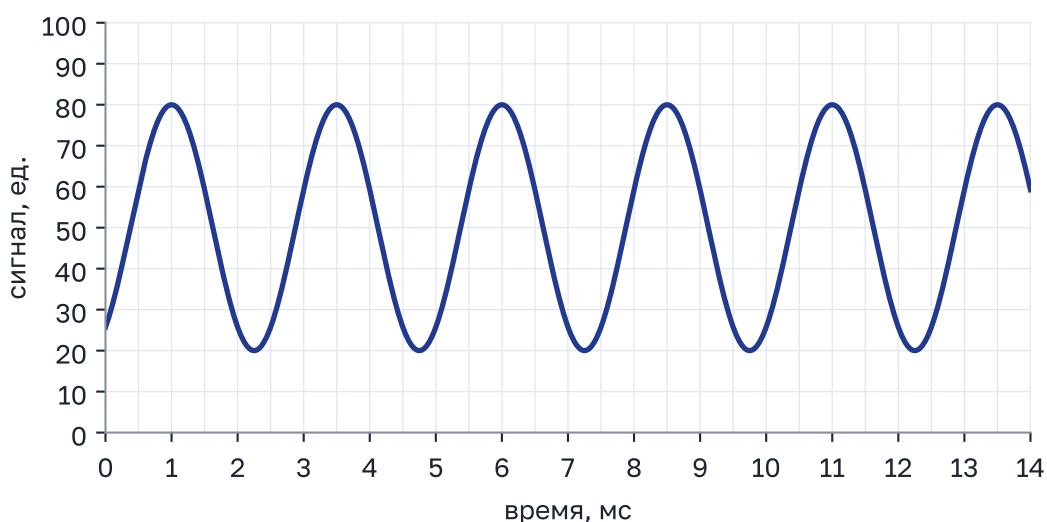


Рисунок 2.12 – Сигнал с микрофона. По горизонтали – время в миллисекундах, одна клеточка – 0,5 мс

Дано: график сигнала.

Найти: A – ?, f – ?

Решение

Шаг 1. Оси. По горизонтали время в миллисекундах, крупные деления через 1 мс, тонкие линии – через 0,5 мс. По вертикали – значение сигнала в условных единицах (ед.).

Шаг 2. Наибольшее и наименьшее. Все гребни доходят до 80, все впадины – до 20.

Шаг 3. Средний уровень и амплитуда. Средний уровень $(80 + 20) : 2 = 50$. Амплитуда – от среднего уровня до гребня: $A = 80 - 50 = 30$ ед. Или через размах: $(80 - 20) : 2 = 30$ ед.

Шаг 4. Гребни подальше друг от друга. Первый гребень – в 1 мс. Дальше гребни в 3,5; 6; 8,5; 11 и 13,5 мс. Возьмём первый и последний: между ними $13,5 - 1 = 12,5$ мс. Сколько там колебаний? Гребней шесть, а промежутков между ними – пять.

Шаг 5. Период. $T = 12,5 : 5 = 2,5$ мс. Проверим по соседним гребням: $3,5 - 1 = 2,5$ мс – то же самое.

Шаг 6. Период в секунды и частота. $T = 2,5 : 1000 = 0,0025$ с. $f = 1 : 0,0025 = 400$ Гц.

Прикинем по-другому: за 10 мс на графике проходит 4 колебания (гребни в 1; 3,5; 6; 8,5 и снова через 10 мс после первого, в 11). Сотая доля секунды – 4 колебания, целая секунда – в 100 раз больше, 400.

⚠ Типичная ошибка. В амплитуде – взять верхнюю точку целиком и ответить «80». Амплитуду отсчитывают от среднего уровня, а он здесь 50, а не 0. Другая ошибка – записать в ответ размах 60. В частоте чаще всего путают гребни и промежутки: шесть гребней – это пять периодов, а не шесть. С шестью выйдет $12,5 : 6 = 2,08$ мс и 480 Гц.

Ответ: а) 30 ед.; б) 400 Гц.

Своя задача по программе сферы, модуль 2

Реши сам

2.3. На экране прибора уложились 9 полных колебаний сигнала, и на них ушло 36 мс. Найди частоту сигнала в герцах.

2.4. Датчик освещённости стоит перед мигающей гирляндой. Его сигнал качается между 140 и 260 ед., гребни приходятся на 0,3 с; 0,8 с; 1,3 с и 1,8 с. Найди амплитуду сигнала и частоту мигания в герцах.

Задача 2.3 Сколько отсчётов

дискретизация, частота отсчётов, отсчёты на период

Программа на ноутбуке записывает звук с микрофона с частотой дискретизации 8 кГц. Миша записал ноту, которую сыграл на синтезаторе: она звучала 2,5 с, её частота – 400 Гц. а) Сколько отсчётов получилось в записи? б) Сколько отсчётов приходится на одно колебание звука?

Дано: $f_d = 8$ кГц; $t = 2,5$ с; $f = 400$ Гц.

Найти: N – ?, отсчётов на период – ?

Решение

Шаг 1. Килогерцы в герцы. $8 \text{ кГц} = 8 \cdot 1000 = 8000$ Гц. Значит, каждую секунду программа делает 8000 отсчётов.

Шаг 2. Отсчётов в записи. $N = f_d \cdot t = 8000 \cdot 2,5 = 20\,000$.

Шаг 3. Отсчётов на период. За секунду – 8000 отсчётов и 400 колебаний ноты. На одно колебание: $8000 : 400 = 20$ отсчётов.

Можно через время. Шаг между отсчётами $\Delta t = 1 : 8000 = 0,000125$ с = 0,125 мс. Период ноты $T = 1 : 400 = 0,0025$ с = 2,5 мс. В одном периоде $2,5 : 0,125 = 20$ шагов.

Двадцать отсчётов на колебание – с запасом больше двух, так что форму ноты по точкам видно хорошо.

❗ **Типичная ошибка.** Забыть про «кило» и посчитать $8 \cdot 2,5 = 20$ отсчётов на всю запись. Двадцать измерений за две с половиной секунды – это меньше, чем колебаний у ноты, звук по ним не восстановить. Во втором вопросе иногда делят наоборот: $400 : 8000 = 0,05$. Меньше одного отсчёта на колебание быть не может, если частота дискретизации больше частоты звука, – значит, перепутали, что на что делить.

Ответ: а) 20 000 отсчётов; б) 20 отсчётов.

Своя задача по программе сферы, модуль 2

Реши сам

2.5. Песню длиной 3 минуты записали на компакт-диск с частотой дискретизации 44 100 Гц. Сколько отсчётов получилось в одном канале?

2.6. Инженер хочет, чтобы на каждое колебание сигнала 500 Гц приходилось не меньше 10 отсчётов. Какая наименьшая частота дискретизации ему нужна? Ответ в герцах.

Задача 2.4 КиберМишка считает хлопки

детектирование по уровню, порог, переход снизу вверх

КиберМишка слушает микрофоном и считает хлопки. Громкость он измеряет каждые 50 мс, первый отсчёт – в момент 0 мс. Хлопок засчитывается, когда громкость переходит через порог 60 снизу вверх: предыдущий отсчёт меньше 60, а следующий – 60 или больше. Вот 20 отсчётов по порядку:

12, 15, 11, 74, 88, 41, 14, 10, 13, 16, 67, 72, 23, 12, 9, 18, 91, 85, 30, 11.

Сколько хлопков насчитает КиберМишка и в какой момент он засчитает первый? В ответ введи два числа через пробел: число хлопков и время первого в миллисекундах.

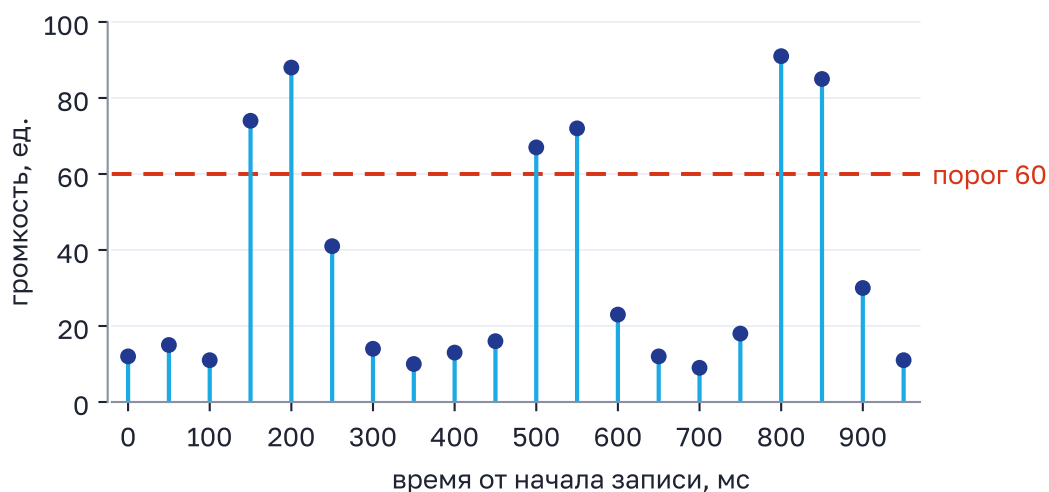


Рисунок 2.13 – Отсчёты громкости и порог 60

Дано: 20 отсчётов; шаг $\Delta t = 50$ мс; порог 60; срабатывание – переход снизу вверх.

Найти: число хлопков; время первого, мс.

Решение

Шаг 1. Нули и единицы. Под каждым отсчётом пишем 1, если он 60 или больше, и 0, если меньше:

0, 0, 0, **1**, **1**, 0, 0, 0, 0, 0, 0, **1**, **1**, 0, 0, 0, 0, **1**, **1**, 0, 0.

Шаг 2. Переходы 0 → 1. Их три: $11 \rightarrow 74$, $16 \rightarrow 67$, $18 \rightarrow 91$. Значит, хлопков три. Каждый хлопок звучал два отсчёта подряд, но засчитан один раз.

Шаг 3. Номер первого перехода. Нумеруем отсчёты с нуля: 12 – номер 0, 15 – номер 1, 11 – номер 2, 74 – номер 3. Хлопок засчитан на отсчёте номер 3.

Шаг 4. Время. $3 \cdot 50 = 150$ мс.

На рисунке это легко проверить: первая точка выше красной линии стоит посередине между отметками 100 и 200 мс. Остальные хлопки – в $10 \cdot 50 = 500$ мс и $16 \cdot 50 = 800$ мс. Между хлопками по 0,3–0,35 с – человек хлопает как раз примерно так.

Проверим кодом. Строки после # – пояснения, Python их не читает.

```
громкость = [12, 15, 11, 74, 88, 41, 14, 10, 13, 16,
              67, 72, 23, 12, 9, 18, 91, 85, 30, 11]
порог = 60
хлопки = []
for i in range(1, len(громкость)): # i – номер отсчёта: 1, 2, ..., 19
    if громкость[i - 1] < порог ≤ громкость[i]: # был ниже, стал не ниже
        хлопki.append(i * 50) # время = номер · шаг
print(len(хлопки), хлопki) # 3 [150, 500, 800]
```

Первая строка кладёт отсчёты в **список** – пронумерованный ряд чисел; `громкость[3]` – это отсчёт номер 3, то есть 74. `for` перебирает номера, начиная с 1: у нулевого отсчёта нет предыдущего. `if` проверяет переход. Запись `a < порог ≤ b` Python читает как «а меньше порога, а порог не больше b» – две проверки сразу. `append` дописывает момент в конец списка `хлопки`.

❗ **Типичная ошибка.** Посчитать все точки выше порога и ответить «6 хлопков». Пятна громкости длиннее одного отсчёта, поэтому считают переходы. Во втором числе часто пишут 200 мс: 74 – четвёртое число в ряду, и его умножают на 4. Но первое число сделано в момент 0, так что четвёртое – через три шага, в 150 мс.

Ответ: 3 150.

Своя задача по программе сферы, модуль 2

Реши сам

2.7. На даче к крышке почтового ящика прикрепили датчик вибрации. Когда в ящик бросают письмо, сигнал подскакивает. Порог 6, срабатывание – когда предыдущий отсчёт меньше 6, а следующий – 6 или больше. Отсчёты: 3, 8, 9, 2, 7, 10, 4, 1, 9, 5. Сколько раз сработал датчик?

2.8. В классе шумно, и КиберМишку из задачи 2.4 настроили строже: хлопок засчитывается, только если после отсчёта ниже 60 идут **два отсчёта подряд** 60 или больше. Отсчёты: 10, 14, 65, 12, 11, 78, 83, 20, 13, 70, 9, 15, 81, 76, 88, 17. а) Сколько хлопков насчитал бы Мишка по старому правилу, с одним отсчётом? б) Сколько – по новому? В ответ введи два числа через пробел.

Задача 2.5 Велокомпьютер

детектирование по периоду, частота вращения, скорость

На спице велосипедного колеса закреплён магнитик, на вилке – датчик. Каждый раз, когда магнитик пролетает мимо, датчик даёт щелчок. Окружность колеса – 2 м. Велокомпьютер записал пять щелчков подряд в моменты 0,3 с; 0,7 с; 1,1 с; 1,5 с и 1,9 с. С какой скоростью едет велосипед? Ответ дай в км/ч.

Дано: щелчки в 0,3; 0,7; 1,1; 1,5; 1,9 с; окружность $L = 2$ м.

Найти: v в км/ч – ?

Решение

Шаг 1. Что значит щелчок. Щелчок – один полный оборот колеса. Время между соседними щелчками – период оборота.

Шаг 2. Период. От первого щелчка до последнего $1,9 - 0,3 = 1,6$ с. Щелчков пять, промежутков – четыре. $T = 1,6 : 4 = 0,4$ с. Соседние щелчки подтверждают: $0,7 - 0,3 = 0,4$ с.

Шаг 3. Частота вращения. $f = 1 : 0,4 = 2,5$ оборота в секунду.

Шаг 4. Скорость. За один оборот велосипед проезжает окружность колеса, 2 м. За секунду – $2 \cdot 2,5 = 5$ м. Скорость $v = 5$ м/с.

Шаг 5. В км/ч. Из главы 1: чтобы перевести м/с в км/ч, умножаем на 3,6. $5 \cdot 3,6 = 18$ км/ч.

Восемнадцать километров в час – спокойная езда по велодорожке: быстрее, чем бегом, но до гонки далеко.

 **Типичная ошибка.** Разделить 1,6 с на пять щелчков: $T = 0,32$ с, и выйдет 22,5 км/ч. Пять щелчков дают четыре промежутка, как у забора из пяти столбиков – четыре пролёта. Ещё можно забыть последний перевод и ответить «5». Но спрашивали км/ч, а 5 – это метры в секунду.

Ответ: 18 км/ч.

Своя задача по программе сферы, модуль 2

Реши сам

2.9. На колесе самоката окружностью 0,5 м стоит такой же датчик: один щелчок за оборот. Между соседними щелчками 0,25 с. С какой скоростью едет самокат? Ответ в м/с.

2.10. Датчик стоит у вентилятора с четырьмя лопастями и щёлкает каждый раз, когда мимо проходит лопасть. Щелчки идут каждые 5 мс. Сколько оборотов в минуту делает вентилятор?

Задача 2.6* Два микрофона

амплитуда по таблице, период, задержка звука, скорость звука

На тренировке по акустическому каналу динамик играет ровный тон, а два микрофона записывают его. Каждые 0,5 мс приёмник сохраняет по числу с каждого микрофона. Второй микрофон стоит дальше от динамика, чем первый, но не больше чем на 1 м. Скорость звука в воздухе – 340 м/с. Вот начало записи:

Таблица 2.1

Время, мс	Микрофон 1	Микрофон 2
0	512	432
0,5	653	455
1,0	712	512
1,5	653	569
2,0	512	592
2,5	371	569
3,0	312	512
3,5	371	455
4,0	512	432
4,5	653	455
5,0	712	512
5,5	653	569
6,0	512	592

Найди: а) амплитуду сигнала первого микрофона; б) частоту тона в герцах; в) на сколько сантиметров второй микрофон дальше от динамика, чем первый. В ответ введи три числа через пробел.

Дано: таблица отсчётов, шаг 0,5 мс; разница расстояний не больше 1 м; $v = 340$ м/с.

Найти: A_1 , f , разницу расстояний в см.

Решение

Шаг 1. Амплитуда первого микрофона. Наибольшее значение в столбце – 712, наименьшее – 312. Средний уровень $(712 + 312) : 2 = 512$. Амплитуда $A_1 = (712 - 312) : 2 = 200$.

Шаг 2. Период. Гребни первого микрофона (значение 712) – в 1,0 мс и в 5,0 мс.

$T = 5,0 - 1,0 = 4$ мс. У второго микрофона гребни (592) – в 2,0 мс и 6,0 мс, период тот же, 4

мс. Оба микрофона слушают один тон, поэтому периоды и совпали.

Шаг 3. Частота. $T = 4 \text{ мс} = 0,004 \text{ с}$, $f = 1 : 0,004 = 250 \text{ Гц}$.

Шаг 4. Задержка. Второй микрофон дальше, значит, звук приходит к нему позже. Гребень, который первый микрофон поймал в 1,0 мс, у второго появится в 2,0 мс.



Рисунок 2.14 – Звук от динамика сначала доходит до первого микрофона, потом до второго

Задержка $1 \text{ мс} = 0,001 \text{ с}$. Здесь есть ловушка: все гребни одинаковые, и по ним не видно, какой из них «тот же». Может, гребень из 1,0 мс дошёл до второго микрофона только следующим, в 6,0 мс? Тогда задержка 5 мс, и звук прошёл бы лишние $340 \cdot 0,005 = 1,7 \text{ м}$. А по условию разница не больше 1 м. Значит, задержка 1 мс.

Шаг 5. Разница расстояний. Лишний путь до второго микрофона звук проходит за 1 мс:
 $s = v \cdot t = 340 \cdot 0,001 = 0,34 \text{ м} = 34 \text{ см}$.

Амплитуда второго микрофона $(592 - 432) : 2 = 80$ – он слышит тише, как и положено дальнему. А треть метра разницы легко уместится на обычной парте.

⚠ Типичная ошибка. В пункте а) ответить «712» или «400». 712 – это отсчёт от нуля, а у этих микрофонов тишина записывается числом 512; 400 – размах, он вдвое больше амплитуды. В пункте в) забывают перевести миллисекунды в секунды: $340 \cdot 1 = 340$ «метров» – это уже другой конец города. И не забудь, что спрашивали сантиметры.

Ответ: 200 250 34.

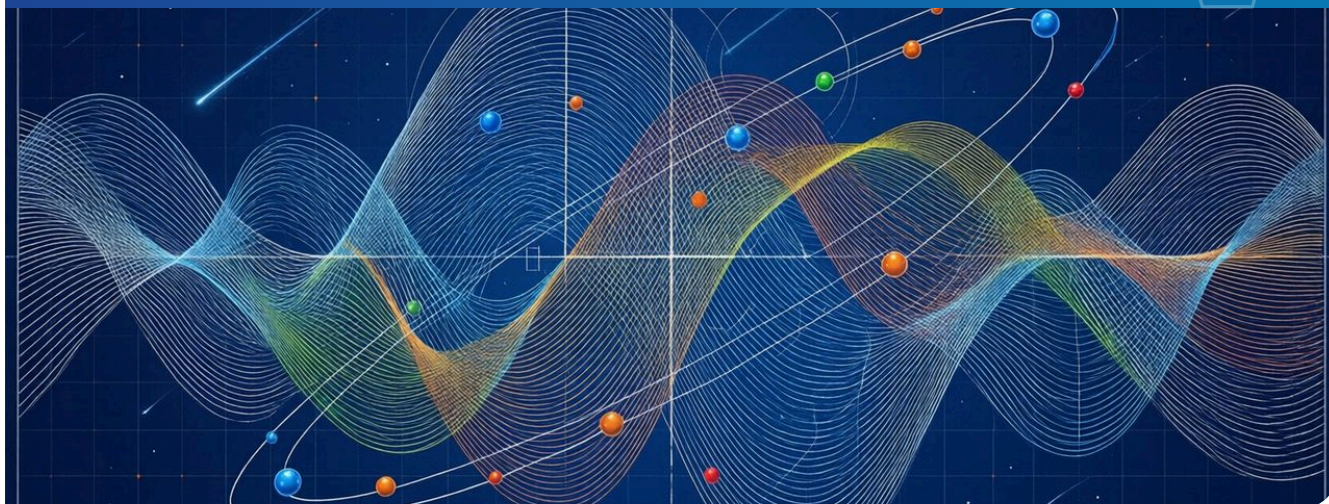
По мотивам задачи первого отборочного этапа НТО 2025/26, профиль «Технологии беспроводной связи», «Радиоразминка»

Реши сам

2.11. Два микрофона записали один и тот же звук. У первого сигнал качается от 190 до 830, у второго – от 410 до 610. Во сколько раз амплитуда первого больше амплитуды второго?

2.12.* Динамик играет ровный тон. Гребни сигнала первого микрофона приходятся на 1,0; 6,0 и 11,0 мс, второго – на 3,0; 8,0 и 13,0 мс. Второй микрофон стоит дальше от динамика, но не больше чем на 1 м. Скорость звука – 340 м/с. Найди частоту тона в герцах и на сколько сантиметров второй микрофон дальше первого. В ответ введи два числа через пробел.

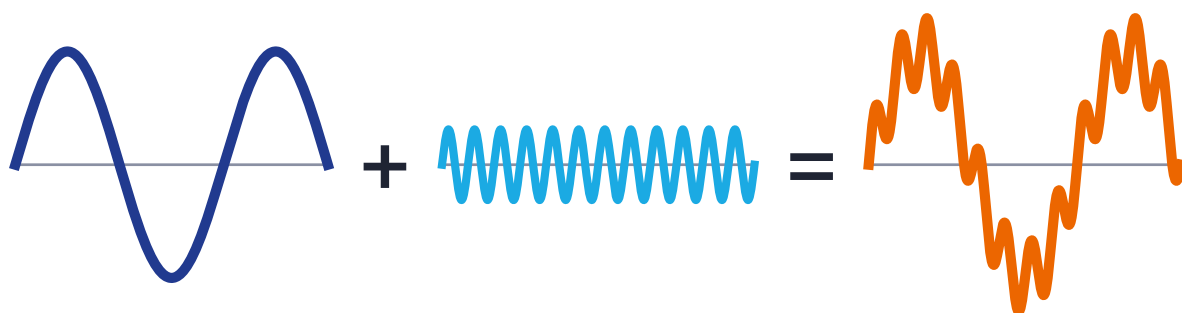
Разложение сигнала: спектр и фильтры



Сигнал с настоящего датчика мало похож на красивую волну из учебника. Он дрожит и скачет, медленное в нём перемешано с быстрым, а иногда прилетает число, которого там быть не должно. Эта глава учит разбирать такую «кашу» на части и отделять полезное от мусора. Считать почти не придётся: хватит сложения, деления и внимательного глаза. Ответ в таких задачах – число или строка чисел.

3.1. Сложная волна – сумма простых

НА ПАЛЬЦАХ



медленная большая быстрая мелкая вместе

Рисунок 3.1 – Медленная большая волна плюс быстрая мелкая – получается волна с зазубринами

Сложи две простые волны – выйдет сложная. Работает и в обратную сторону: любую сложную волну можно разобрать на простые.

В главе 2 мы разбирали самую простую волну – **синусоиду**. Она плавно качается вверх-вниз, как качели, и её описывают два числа: **амплитуда** (на сколько волна отходит от середины) и **частота** (сколько колебаний в секунду). Так звучит камертон – металлическая вилочка, по которой настраивают гитару: чистое ровное «у-у-у».

Теперь включи на телефоне диктофон и скажи «привет». На экране появится не гладкая волна, а зубчатый забор. Голос, музыка, гул мотора, дрожь датчика – всё это **сложные сигналы**. Двести лет назад французский математик Жозеф Фурье показал: такой сигнал можно собрать из простых синусоид, у каждой – своя частота и своя амплитуда. Разбор сигнала на простые волны называют **декомпозицией**, по-русски – разложением. Почти всё в этой главе вырастает из этой мысли.

Как складывают волны. Складывают по точкам. Берём один момент времени и смотрим, где в этот момент первая волна и где вторая. Складываем два числа – получаем точку суммы. Повторяем для всех моментов, соединяем точки – готова новая линия.

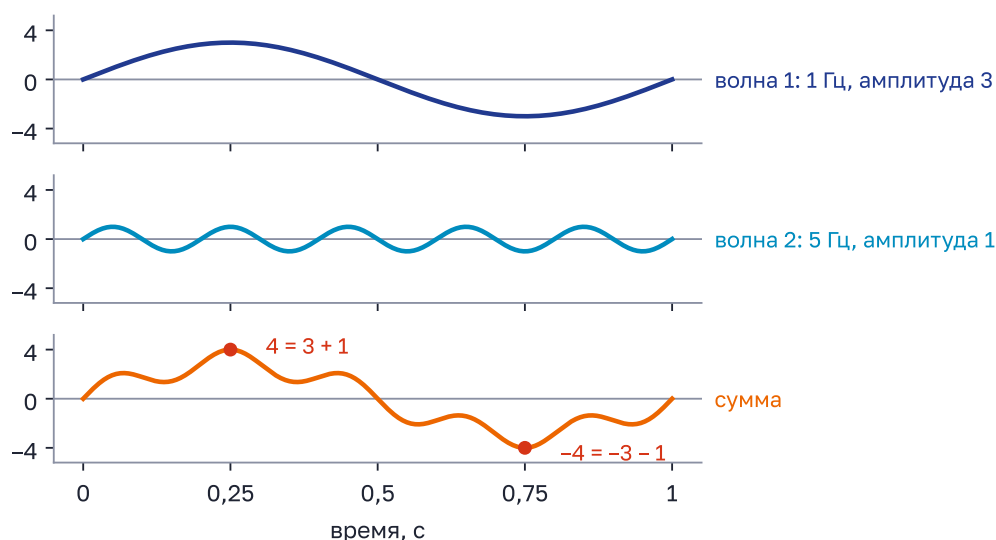


Рисунок 3.2 – Сложение двух волн. В момент 0,25 с обе волны наверху, и сумма поднимается до $3 + 1 = 4$

Несколько точек с этого рисунка (значения сняты с графика и округлены до десятых):

Таблица 3.1

Время, с	Волна 1	Волна 2	Сумма
0,05	0,9	1	1,9
0,25	3	1	4
0,35	2,4	-1	1,4
0,5	0	0	0
0,75	-3	-1	-4

В момент 0,25 с обе волны на вершине, и сумма поднимается выше всего: $3 + 1 = 4$. А в 0,35 с первая волна ещё наверху, вторая уже внизу, и они частично гасят друг друга: $2,4 + (-1) = 1,4$. Отсюда полезное правило: сумма никогда не поднимется выше, чем сумма амплитуд. Для двух волн это $A_1 + A_2$, для трёх – $A_1 + A_2 + A_3$. Ниже минус суммы амплитуд она тоже не опустится.

Запомни. Волны складывают по точкам: в каждый момент значение суммы равно сумме значений. Потолок сложной волны – сумма амплитуд, пол – та же сумма с минусом.

Для любопытных. Сложение волн легко повторить в электронной таблице. В столбец A запиши моменты времени от 0 до 1 с шагом 0,01. В B2 – формулу $=3*\text{SIN}(2*\text{ПИ}()*1*A2)$ (волна 1 Гц с амплитудой 3), в C2 – $=1*\text{SIN}(2*\text{ПИ}()*5*A2)$, в D2 – $=B2+C2$. Протяни формулы вниз и построй график трёх столбцов. Поменяй частоты и амплитуды – и посмотри, как меняется сумма.

3.2. Спектр – «рецепт» сигнала

НА ПАЛЬЦАХ

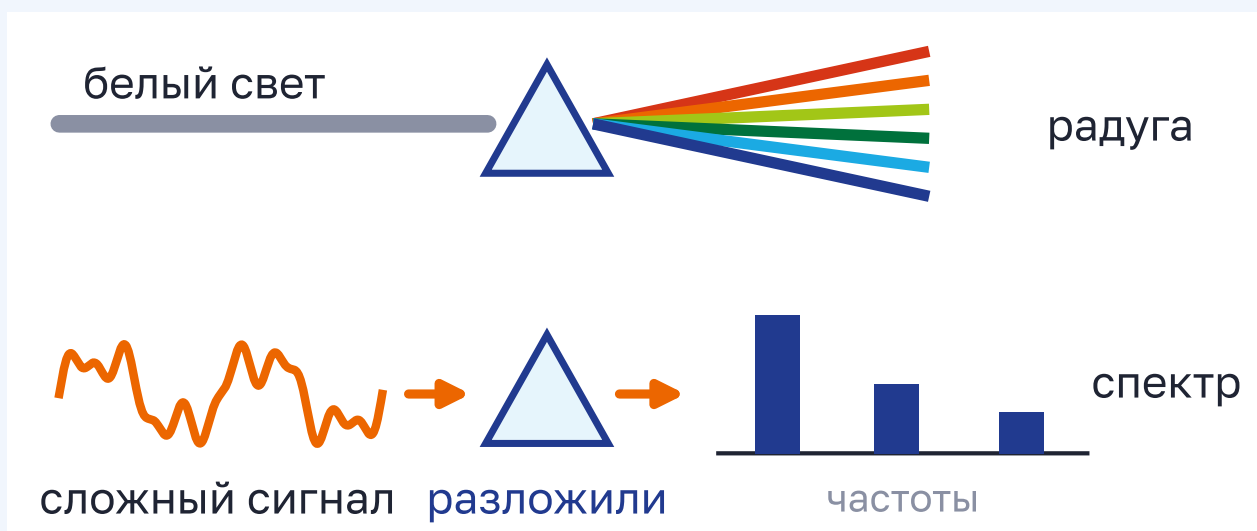


Рисунок 3.3 – Призма раскладывает белый свет на цвета радуги. Спектр так же раскладывает сигнал на простые волны

Каждый столбик – одна простая волна. Где столбик стоит – её частота, какой он высоты – её амплитуда.

Исаак Ньютон пропустил солнечный луч через стеклянную призму, и на стене появилась разноцветная полоска. Её назвали **спектром**. Оказалось, что белый свет – смесь цветов, и призма разводит их в стороны. Со сложным сигналом можно сделать то же самое: разложить его по частотам и посмотреть, из каких простых волн он сложен.

Похоже на борщ. По вкусу не скажешь, сколько в нём свёклы и сколько капусты. А в рецепте всё написано: свёкла – 300 г, капуста – 200 г, морковь – 100 г. **Спектр** – такой же рецепт сигнала: какие простые волны в него положили и сколько каждой.

У одного и того же сигнала есть два портрета:

- **осциллограмма** – график «сигнал во времени». По горизонтали время, по вертикали значение сигнала. Такой график рисует **осциллограф** – прибор, который показывает, как меняется напряжение;

- **спектр** – график «сигнал по частотам». По горизонтали частота в герцах, по вертикали амплитуда. Каждая простая волна – один столбик.



Рисунок 3.4 – Один сигнал двумя способами. Сверху – как он меняется во времени, снизу – из каких волн он сложен: 100 Гц с амплитудой 3, 200 Гц с амплитудой 1 и 350 Гц с амплитудой 1,5

По осциллограмме сверху не угадаешь, что внутри три волны: линия просто петляет. Спектр снизу отвечает сразу. Три столбика – три волны. Столбик над 100 Гц самый высокий, значит, волна 100 Гц самая сильная. Волна 350 Гц слабее её вдвое: $3 : 1,5 = 2$.

Как читать спектр. Сначала подписи осей: что по горизонтали, что по вертикали, в каких единицах. Потом шаг сетки: если подписи стоят через 200 Гц, а между ними есть ещё чёрточки, то чёрточки идут через 100 Гц. Столбик, стоящий между подписями, придётся «досчитать» по чёрточкам. Теперь пересчитай столбики – столько в сигнале простых волн. Под каждым столбиком внизу читаешь частоту его волны, а по высоте – амплитуду.

Основная частота и обертоны. Нота ля первой октавы – это 440 Гц: на эту частоту настраивают камертоны. Но гитара, пианино и флейта, играя одну и ту же ля, звучат по-разному. Дело в спектре. Кроме основной волны 440 Гц в звуке есть более тихие волны на 880 Гц (вдвое выше), 1320 Гц (втрое выше) и так далее. Их называют **обертонами** – «верхними тонами». Какие обертоны есть и насколько они громкие, у каждого инструмента своё, и от этого зависит **тембр** – «окраска» звука. А у шума (шипение воды, шорох листьев) основной волны нет вовсе: спектр похож на траву – много низеньких столбиков на всех частотах.

Встретишь ещё слово **гармоники**. Так называют основную волну вместе с обертонами, и счёт у гармоник начинается с основной: 440 Гц – первая гармоника, 880 Гц – вторая. Получается, 880 Гц – это вторая гармоника и в то же время первый обертон. Номера сдвинуты на единицу, не перепутай.

Зачем спектр инженеру? По спектру вибрации двигателя видно поломку: в нём появляется столбик, которого у исправного мотора не было. Орнитологи – учёные, которые изучают птиц, – узнают птицу по спектру её песни, как человека по отпечатку пальца. А ползунки эквалайзера в музыкальном плеере поднимают и опускают отдельные частоты спектра: подвинул «басы» вверх – низкие столбики выросли.

Запомни. Осциллограмма – сигнал во времени, спектр – сигнал по частотам. На спектре один столбик – одна простая волна: место столбика – частота, высота – амплитуда. Обертоны музыкального звука стоят на частотах, кратных основной: в 2, 3, 4 раза выше.

Для любопытных. Посмотри на осциллограмму ещё раз: узор от 0 до 20 мс повторяется от 20 до 40 мс. Частоты 100, 200 и 350 Гц все делятся на 50, и 50 – наибольшее число, на которое они делятся все. Поэтому сумма повторяется 50 раз в секунду, то есть каждые $1 : 50 = 0,02 \text{ с} = 20 \text{ мс}$.

3.3. Шум и фильтр низких частот

НА ПАЛЬЦАХ

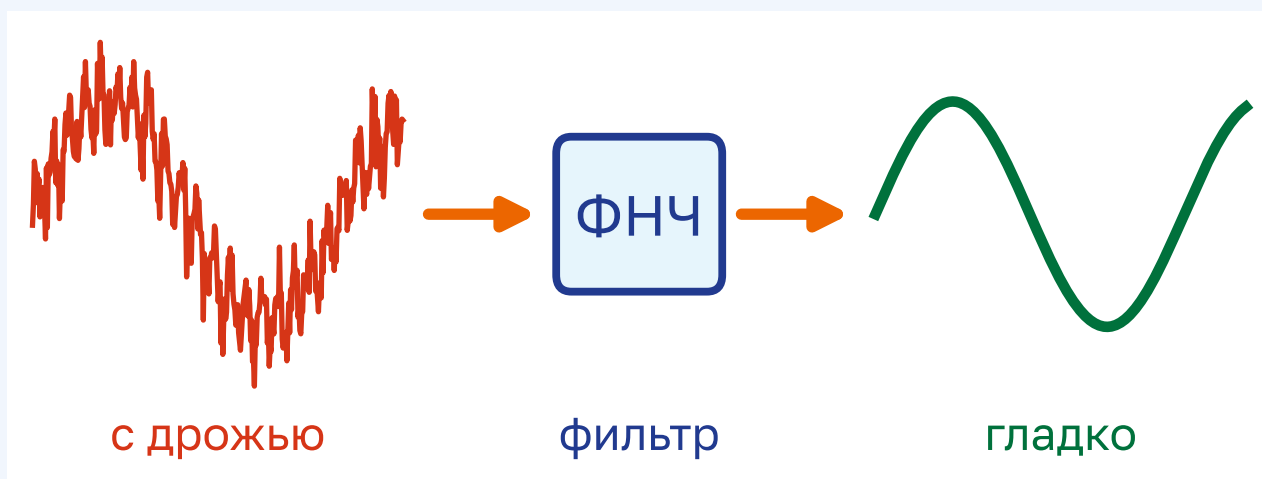


Рисунок 3.5 – Фильтр низких частот пропускает медленные колебания и задерживает быструю дрожь

На входе сигнал был похож на пилу с зазубринами, на выходе он гладкий: осталась одна медленная волна.

Шум, или помехи, – всё, что примешалось к полезному сигналу по дороге: тряска мотора, наводки от соседних проводов, случайные ошибки датчика. Возьмём робот-пылесос. Он едет вдоль стены, и расстояние до неё меняется плавно. Но колёса подпрыгивают на стыках ламината, и датчик выдаёт 20, 23, 18, 22 см. Если верить каждому числу, робот решит, что стена то подходит, то убегает, и начнёт дёргаться из стороны в сторону.

Вспомним спектр. Плавное изменение расстояния – это медленные волны, у них **низкая частота**: одно-два колебания в секунду. Дрожь от колёс – быстрые волны, у них **высокая частота**: десятки и сотни колебаний в секунду. Полезное и вредное живут в разных частях спектра. Значит, их можно разделить по частоте.

Фильтр низких частот (ФНЧ) пропускает низкие частоты и задерживает высокие. Граница между ними называется **частотой среза**. Всё, что ниже неё, проходит, всё, что выше, гасится. На картинке спектра ФНЧ стирает все столбики правее частоты среза. У настоящего фильтра граница слегка размыта: волны рядом со срезом ослабевают постепенно. В задачах книги,

если не сказано другое, фильтр считаем идеальным: волна ниже среза проходит целиком, выше – пропадает.

Фильтр высоких частот (ФВЧ) работает наоборот: убирает медленное и оставляет быстрое. Он нужен, когда полезное как раз быстрое. Сканер штрих-кода на кассе видит свет, отражённый от полосок: чёрная – темно, белая – светло, перепады очень частые. А общее освещение в зале меняется медленно, особенно если кассир ведёт сканер неровно. ФВЧ убирает этот медленный фон, и остаются только полоски.

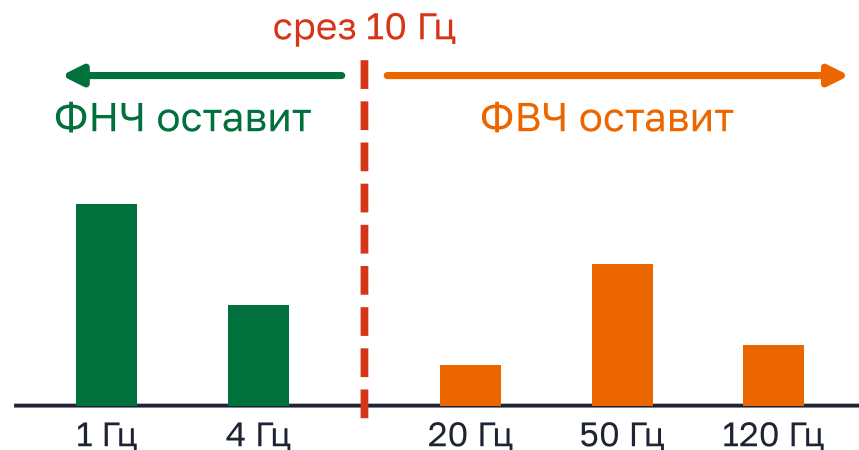


Рисунок 3.6 – Один спектр и одна частота среза 10 Гц. Всё, что левее красной черты, пропустит ФНЧ, всё, что правее, – ФВЧ

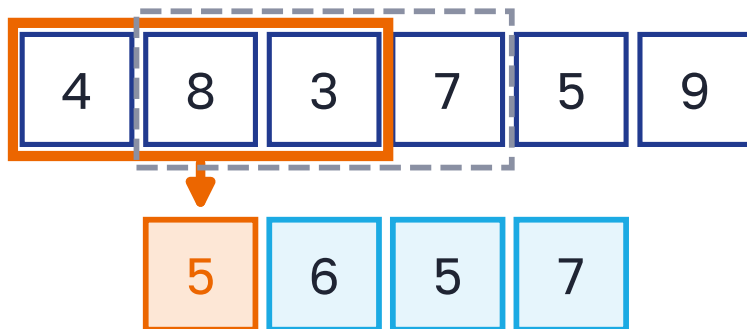
Где стоит ФНЧ? Почти везде, где есть датчик. У квадрокоптера датчик наклона трясётся от моторов, и без фильтра автопилот сошёл бы с ума: ФНЧ стирает тряску, и автопилот видит настоящий наклон. В кардиографе – приборе, который рисует работу сердца зубчатой линией, – ФНЧ убирает помехи от мышц и соседних приборов.

Запомни. Медленное – низкие частоты, быстрое – высокие. ФНЧ пропускает то, что ниже частоты среза, ФВЧ – то, что выше. Фильтр смотрит только на частоту волны; её амплитуда ему не важна.

Совет. Не знаешь, какой фильтр нужен, – спроси себя: «Полезное у меня медленное или быстрое?» Медленное – ставь ФНЧ, быстрое – ФВЧ.

3.4. Скользящее среднее

НА ПАЛЬЦАХ



окно из 3 чисел
шагает вправо

$(4 + 8 + 3) : 3 = 5$,
пишем под серединой

Рисунок 3.7 – Окно из трёх чисел шагает по ряду. Среднее каждого окна записываем под его серединой

Одно большое число в окне делится на троих, поэтому скачки становятся меньше.

На компьютере самый простой фильтр низких частот – **скользящее среднее**. Для него нужно одно умение: находить среднее арифметическое. Среднее нескольких чисел – их сумма, делённая на количество: среднее 4, 8 и 3 равно $(4 + 8 + 3) : 3 = 15 : 3 = 5$.

Скользящее среднее считают так. Берут **окно** – несколько соседних чисел ряда, например три. Находят их среднее и записывают напротив середины окна. Потом сдвигают окно на одно число вправо и считают снова. И так, пока окно не упрётся в конец ряда.

Разберём ряд с картинки: 4, 8, 3, 7, 5, 9, окно 3.

- Первое окно 4, 8, 3: $(4 + 8 + 3) : 3 = 5$. Пишем напротив 8.
- Второе окно 8, 3, 7: $18 : 3 = 6$. Пишем напротив 3.
- Третье окно 3, 7, 5: $15 : 3 = 5$.
- Четвёртое окно 7, 5, 9: $21 : 3 = 7$.

Вышло 5, 6, 5, 7. В исходном ряду соседние числа прыгали на 4–5 единиц (от 8 к 3, от 3 к 7), а в сглаженном меняются на 1–2.

Сколько получится чисел. Из n чисел окном k выходит

$$n - k + 1$$

средних. Здесь $6 - 3 + 1 = 4$. Почему не 6? У краёв ряда полного окна не набрать: у первого числа нет соседа слева, у последнего – справа. С окном 3 теряется по одному числу с каждого края, с окном 5 – по два.

Почему это работает. Дрожь – это быстрые скачки то вверх, то вниз. В одно окно попадают и «плюсы», и «минусы», и они гасят друг друга. А медленное изменение за время одного окна почти не успевает поменяться, и среднее его сохраняет.

Какое окно взять. Чем окно шире, тем линия глаже. Но широкое окно заодно «съедает» то, что было полезным: вершины волн становятся ниже, короткие настоящие события размазываются, а с краёв ряда теряется больше чисел. Узкое окно бережёт детали, зато оставляет часть дрожи. Окна, которое хорошо всем, не бывает: чем-то всегда платишь, а чем можно заплатить – подскажет условие задачи.



Рисунок 3.8 – Одни и те же показания, сглаженные окном 3 и окном 9. Окно 3 убрало мелкие зубцы, но линия ещё неровная. Окно 9 сгладило почти всё, зато вершины волны стали ниже, а с краёв пропало по четыре числа

Запомни. Скользящее среднее: окно из k соседних чисел, их среднее – под середину окна, шаг на одно число. Из n чисел выходит $n - k + 1$ средних. Шире окно – глаже линия, но меньше деталей.

Совет. В электронной таблице пусть числа стоят в столбце В, начиная с клетки В2. В клетку С3 впиши `=СРЗНАЧ(В2:В4)` и протяни формулу вниз за уголок: адреса сдвинутся сами, и каждая клетка С будет стоять напротив середины своего окна. Считаешь вручную – не складывай каждое окно заново: из суммы предыдущего окна вычти число, которое ушло, и прибавь то, которое пришло.

3.5. Несущая и огибающая

НА ПАЛЬЦАХ

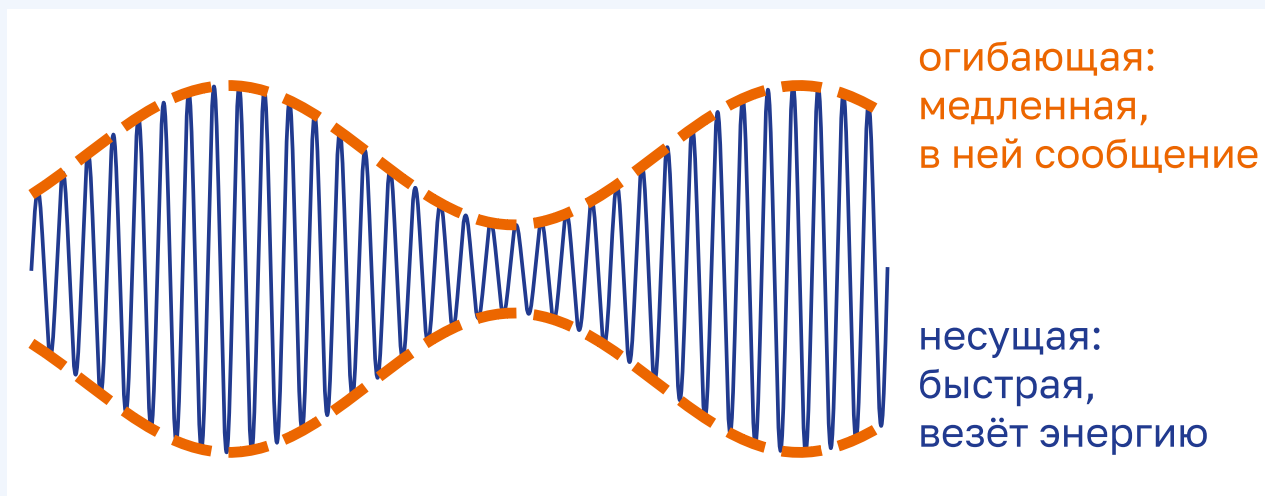


Рисунок 3.9 – Быстрые колебания – несущая. Линия по их вершинам – огибающая

Большой размах, маленький размах – так в сигнал записано сообщение.

Чтобы сигнал улетел далеко – от радиостанции до приёмника на даче, от спутника до Земли, – ему нужна энергия и подходящая частота. За это отвечает **несущая** – быстрое колебание, которое создаёт передатчик. У радиостанций на средних волнах несущая колеблется примерно от полумиллиона до полутора миллионов раз в секунду.

Одна несущая похожа на гудок на одной ноте: слышно, что передатчик включён, и больше ничего. Чтобы передать сообщение, у несущей меняют размах: сильнее – слабее, есть – нет. Проведи линию по верхушкам колебаний – получится **огибающая**, плавная медленная линия. В ней и сидит сообщение: форма звука из студии, команда роботу, точки и тире азбуки Морзе. Как «сажают» сообщение на несущую, разберём в главе 5.

Несущая всегда намного быстрее огибающей: пока огибающая делает одно медленное колебание, несущая успевает качнуться десятки, сотни, а то и тысячи раз. Посчитать это просто. Несущая 1000 Гц, огибающая 10 Гц: за один период огибающей проходит $1000:10 = 100$ колебаний несущей. Если один кусочек сообщения длится 0,02 с, в него уложится $1000 \cdot 0,02 = 20$ колебаний несущей.

Как приёмник достаёт огибающую. В два шага.

- **Выпрямить** – сделать все отрицательные значения положительными: -4 превращается в 4 , а 4 остаётся 4 . В электронной таблице это функция `=ABS(B2)`.
- **Сгладить** выпрямленный сигнал фильтром низких частот, например скользящим средним. Окно берет не короче одного периода несущей.

Зачем выпрямлять? Возьми один период несущей: $0, 4, 0, -4$. Его среднее $(0 + 4 + 0 - 4):4 = 0$. Плюсы и минусы гасят друг друга, и без выпрямления от сообщения остался бы ноль. После выпрямления $0, 4, 0, 4$, и среднее $8:4 = 2$. Чем сильнее колебания, тем больше это число – оно и повторяет огибающую.

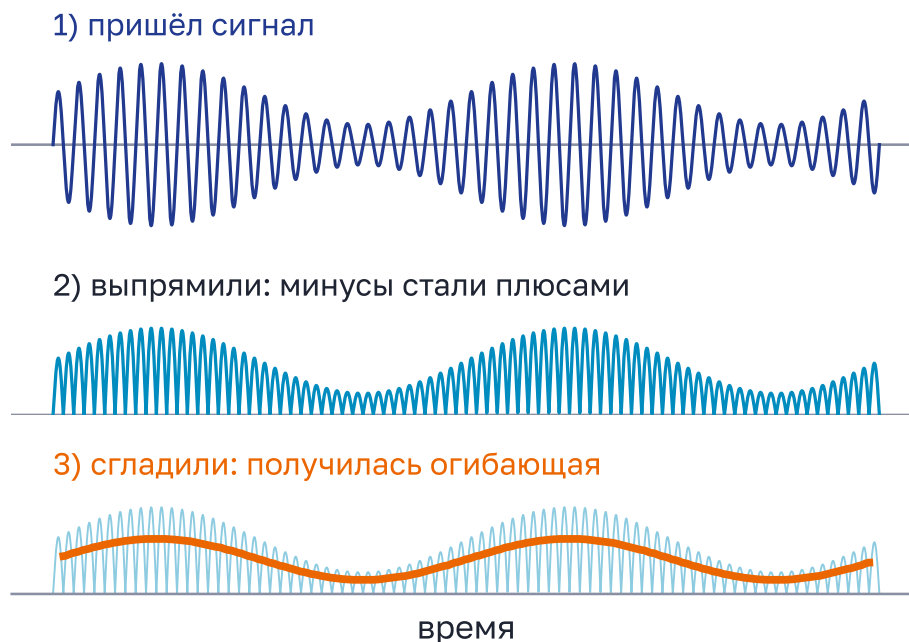


Рисунок 3.10 – Как приёмник выделяет огибающую: выпрямляет сигнал, потом сглаживает его

Заметь: сглаженная линия идёт ниже вершин. Среднее не поднимается выше самого большого числа в окне, а в нашем окне ещё и нули: вышло 2 при амплитуде 4. В настоящем приёмнике сигнал потом усиливают. Для задачи «где колебания сильные, а где слабые» это не мешает: большая огибающая остаётся большой, маленькая – маленькой. Выпрямитель вместе с фильтром называют **амплитудным детектором**.

Где встречается огибающая? Эхолот у рыбака на Обском море посылает в воду короткие пачки ультразвука – звука такого высокого, что человек его не слышит. У каждой пачки своя огибающая. Эхо возвращается, и прибор смотрит на огибающую эха: когда оно пришло и насколько сильное. На заводе так следят за подшипниками – кольцами с шариками, на которых крутится вал станка. По огибающей вибрации видно, что удары внутри нарастают, и станок чинят до поломки.

Запомни. Несущая быстрая и переносит энергию, огибающая медленная и несёт сообщение. Чтобы достать огибающую, сигнал выпрямляют, потом сглаживают. Колебаний несущей в одном кусочке сообщения: частота несущей $\times$ длительность кусочка.

3.6. Аномалия: ищем лишнее

НА ПАЛЬЦАХ



Рисунок 3.11 – Аномалия – значение, которое выбивается из общего ряда

Сначала пойми, как выглядит нормальный ряд. Потом ищи место, где он нарушен.

Аномалия – то, что не похоже на всё остальное: одинокий скачок в ряду показаний, лишний столбик в спектре, сломанный кусочек в повторяющемся сообщении. Бывает, что это ошибка связи – в канале исказилась цифра. Бывает, что это важное событие – в моторе начал разрушаться подшипник. В обоих случаях её нужно заметить, а для этого – с чем-то сравнить.

- **Выброс в ряду чисел** – одинокое значение, которое резко отличается от обоих соседей. Что считать «резко», скажет условие задачи: например, «больше чем на 5». Проверяем каждое число с его левым и правым соседом. Найденный выброс можно заменить средним двух соседей.
- **Лишний столбик в спектре.** Сравниваем спектр «сейчас» со спектром «когда всё было исправно». Новый столбик – новая волна, и у неё есть причина. Особенно подозрительна частота, не кратная основной: у мотора или насоса, который крутится без перебоев, обертоны стоят на частотах в 2, 3, 4 раза выше основной, а чужак выбивается из этого ряда.
- **Нарушенный порядок.** Если маяк повторяет одно и то же сообщение, сначала находим, какой кусок повторяется, потом сверяем с ним каждую копию.

Выброс убирают до сглаживания. Скользящее среднее выброс не стирает – оно размазывает его по соседям. Ряд 2, 2, 2, 29, 2, 2, 2 окном 3 превращается в 2, 11, 11, 11, 2: одно испорченное число испортило три. Проверь сам: $(2 + 2 + 29) : 3 = 33 : 3 = 11$.

Запомни. Аномалию находят сравнением: число – с соседями, спектр – со спектром исправной машины, кусок сообщения – с другими копиями. Выброс чистят до сглаживания, иначе он расплзётся.

Для любопытных. Против одиночных выбросов есть фильтр получше среднего – **медиана** трёх. Три числа из окна расставляют по возрастанию и берут то, что оказалось посередине. Для окна 2, 2, 29 получится 2 – выброс исчез совсем. Ряд из примера выше медианой трёх превращается в 2, 2, 2, 2, 2.

3.7. Разбираем задачи

Задача 3.1 Что гудит в ноутбуке

разминка · спектр, чтение графика

Школьники записали звук вентилятора ноутбука и построили его спектр (рисунок ниже). а) Из скольких простых волн сложен этот звук? б) Какая частота у самой сильной волны? в) У какой волны амплитуда вдвое меньше, чем у самой сильной? В ответ введи три числа через пробел: количество волн, частоту из пункта б) и частоту из пункта в) в герцах.

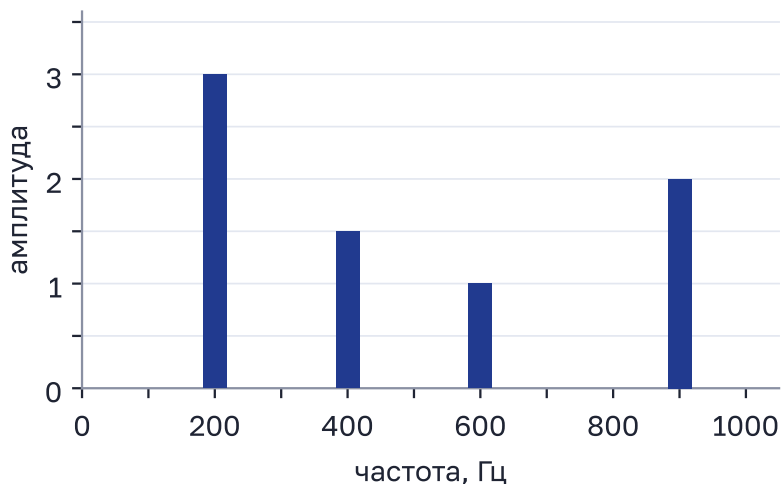


Рисунок 3.12 – Спектр звука вентилятора

Дано: спектр на рисунке.

Найти: число волн; частоту самой сильной волны; частоту волны, которая вдвое слабее.

Решение

Шаг 1. Оси и сетка. По горизонтали – частота в герцах. Подписи стоят через 200 Гц, а чёрточки между ними – через 100 Гц. По вертикали – амплитуда: подписи 1, 2, 3, а линии сетки идут через 0,5.

Шаг 2. Считаем столбики. Их четыре. Каждый столбик – одна простая волна, значит, волн тоже четыре.

Шаг 3. Самая сильная волна. Выше всех столбик высотой 3, он стоит над подписью 200. Частота – 200 Гц.

Шаг 4. Вдвое слабее. Половина от 3 – это $3 : 2 = 1,5$. Ищем столбик, верхушка которого на полпути между линиями 1 и 2. Он стоит над подписью 400 – это 400 Гц.

Для полноты перепишем весь «рецепт»: 200 Гц – 3; 400 Гц – 1,5; 600 Гц – 1; 900 Гц – 2. Последний столбик стоит между подписями 800 и 1000, на чёрточке 900. Обратным ходом: $1,5 \cdot 2 = 3$ – волна 400 Гц действительно вдвое слабее волны 200 Гц.

⚠ Типичная ошибка. Перепутать оси и искать «частоту 1,5» вместо столбика высотой 1,5. Частота всегда читается внизу, под столбиком, а амплитуда – по высоте. Ещё часто приписывают последнему столбику частоту 800 или 1000 Гц: подписи у него нет, и надо до- считать чёрточки.

Ответ: 4 200 400.

Своя задача по программе сферы, модуль 3

Реши сам

3.1. Спектр звука флейты состоит из трёх столбиков: 500 Гц высотой 4, 1000 Гц высотой 2 и 1500 Гц высотой 1. Какие из частот есть в этом звуке: 1) 250 Гц; 2) 500 Гц; 3) 750 Гц; 4) 1000 Гц; 5) 1500 Гц? Введи номера верных вариантов по возрастанию, без пробелов.

3.2. Сигнал сложили из трёх простых волн: 50 Гц с амплитудой 2, 150 Гц с амплитудой 1 и 250 Гц с амплитудой 0,5. а) Сколько столбиков будет на его спектре? б) Какой высоты самый правый столбик? в) Может ли этот сигнал в какой-то момент подняться до 4? В ответ введи через пробел два числа и слово «да» или «нет», например «5 2 да».

Задача 3.2 Дрожит квадрокоптер

разминка · фильтр низких частот, фильтр высоких частот, частота среза

Датчик наклона квадрокоптера записал сигнал, и его разложили на спектр. В спектре пять волн:

1. 2 Гц, амплитуда 6 – квадрокоптер наклоняется по командам пилота;
2. 8 Гц, амплитуда 2 – его раскачивает ветер;
3. 40 Гц, амплитуда 0,5 – гудит рама;
4. 150 Гц, амплитуда 3 – трясутся моторы;
5. 300 Гц, амплитуда 1 – дрожат винты.

а) Сигнал пропустили через фильтр низких частот с частотой среза 30 Гц. Какие волны останутся? б) Какие волны остались бы, если вместо него поставить фильтр высоких частот с той же частотой среза? Фильтры идеальные: волна либо проходит целиком, либо пропадает. В ответ введи номера волн для а) и для б) через пробел, каждую группу – по возрастанию и без пробелов внутри (например, «14 235»).

Дано: пять волн с частотами 2, 8, 40, 150 и 300 Гц; частота среза 30 Гц.

Найти: какие волны пройдут через ФНЧ и через ФВЧ.

Решение

Шаг 1. Что делает ФНЧ. Пропускает всё, что ниже частоты среза, и гасит всё, что выше. Сравниваем с 30 Гц каждую частоту:

- 2 Гц меньше 30 – проходит;
- 8 Гц меньше 30 – проходит;
- 40 Гц больше 30 – гасится;
- 150 Гц и 300 Гц – тем более гасятся.

Через ФНЧ пройдут волны 1 и 2.

Шаг 2. Что делает ФВЧ. Наоборот: пропускает то, что выше 30 Гц. Это волны 3, 4 и 5.

Шаг 3. Проверяем. Каждая волна попала только в один из двух ответов, а вместе они дают все пять: 1, 2 и 3, 4, 5. У пары фильтров с одним срезом иначе и не бывает. Теперь взгляни глазами автопилота: после ФНЧ он видит команды пилота и порывы ветра – как раз то, с чем ему надо бороться, а тряска моторов и винтов ушла.

 **Типичная ошибка.** Смотреть на амплитуду: «волна 150 Гц сильная, амплитуда 3, – она пройдёт». Фильтру всё равно, насколько волна сильная, он сортирует волны только по частоте. Бывает и другая путаница: «низких частот» понимают как «пропускает низкие амплитуды». Нет, речь о частоте – о том, как быстро волна качается.

Ответ: 12 345.

Своя задача по программе сферы, модуль 3

Реши сам

3.3. Микрофон записал звук на кухне. В спектре пять волн: 1) 100 Гц – гудит холодильник; 2) 440 Гц – кто-то поёт ноту ля; 3) 880 Гц – обертон этой ноты; 4) 3000 Гц – пищит микроволновка; 5) 8000 Гц – шипит вода в кране. Звук пропустили через ФНЧ с частотой среза 1000 Гц. Введи номера волн, которые пройдут, по возрастанию, без пробелов.

3.4. Тот же звук из задачи 3.3 пропустили сначала через ФНЧ с частотой среза 1000 Гц, а потом через ФВЧ с частотой среза 300 Гц. Какие волны остались? Введи их номера по возрастанию, без пробелов.

Задача 3.3 Робот-пылесос подъезжает к стене

скользящее среднее, фильтр низких частот

Робот-пылесос медленно подъезжает к стене. Датчик расстояния сделал 10 измерений, в сантиметрах:

61, 57, 59, 61, 54, 56, 58, 51, 53, 52.

Колёса робота трясутся, поэтому показания скачут. Сгладь ряд скользящим средним с окном 3. В ответ введи сглаженный ряд через пробел.

Дано: 10 показаний; окно $k = 3$.

Найти: сглаженный ряд.

Решение

Шаг 1. Сколько чисел получится. $n - k + 1 = 10 - 3 + 1 = 8$. Значит, в ответе будет восемь чисел.

Шаг 2. Считаем окна по очереди. Каждое окно – три соседних числа: складываем их и делим сумму на 3.

- $61 + 57 + 59 = 177, 177:3 = 59;$
- $57 + 59 + 61 = 177, 177:3 = 59;$
- $59 + 61 + 54 = 174, 174:3 = 58;$
- $61 + 54 + 56 = 171, 171:3 = 57;$
- $54 + 56 + 58 = 168, 168:3 = 56;$
- $56 + 58 + 51 = 165, 165:3 = 55;$
- $58 + 51 + 53 = 162, 162:3 = 54;$
- $51 + 53 + 52 = 156, 156:3 = 52.$

Быстрее считать суммы «цепочкой». Из второго окна ушло 61 и пришло 61:

$177 - 61 + 61 = 177$. Из третьего ушло 57, пришло 54: $177 - 57 + 54 = 174$. И так далее.

Шаг 3. Смотрим на результат. 59, 59, 58, 57, 56, 55, 54, 52. Сглаженное расстояние всё время уменьшается, примерно на сантиметр за измерение: робот спокойно подъезжает к стене. А в исходном ряду были «отскоки» $57 \rightarrow 59 \rightarrow 61$, как будто стена отпрыгнула на 4 см. После фильтра их не осталось.

Проверка программой. Если захочешь, то же самое можно посчитать на Python в любом онлайн-интерпретаторе:

```
d = [61, 57, 59, 61, 54, 56, 58, 51, 53, 52] # показания датчика, см
k = 3 # ширина окна
for i in range(len(d) - k + 1): # i - номер первого числа окна
    print(sum(d[i:i + k]) / k, end=" ") # среднее окна
```

Строка 1 – список показаний. Строка 2 – ширина окна. В строке 3 цикл `for` перебирает начало окна: `len(d)` – длина списка (10), так что `i` пробегает от 0 до 7 – это 8 окон. В строке 4 `d[i:i + k]` вырезает три числа, начиная с номера `i`, `sum` их складывает, а `/ k` делит на 3. Программа печатает 59.0 59.0 58.0 57.0 56.0 55.0 54.0 52.0. Python пишет дробную часть через точку.

⚠ Типичная ошибка. Посчитать одно среднее всех десяти чисел: $562:10 = 56,2$. Это одно число, а сглаженный ряд – целая строка. Другая ошибка – потерять последнее окно и написать семь чисел вместо восьми: окно 51, 53, 52 тоже считается, оно просто упирается в конец ряда.

Ответ: 59 59 58 57 56 55 54 52.

Своя задача по программе сферы, модуль 3

Реши сам

3.5. Термометр в теплице на даче каждые полчаса показывал: 18, 21, 15, 24, 18, 21 °С. Сгладь ряд скользящим средним с окном 3. В ответ введи сглаженный ряд через пробел.

3.6. В записи показаний одно число стёрлось: 12, 15, ?, 18. Известно, что после сглаживания окном 3 первое число сглаженного ряда равно 14. Найди стёртое число и второе число сглаженного ряда. Введи их через пробел.

Задача 3.4 Какое окно выбрать

скользящее среднее, ширина окна, компромисс

Робот с датчиком освещённости ждёт команду – вспышку фонарика. Вспышка длится 3 измерения подряд, датчик в это время показывает 9, а до и после вспышки – нули. Ещё бывают блики от окна: одно измерение со значением 3, вокруг тоже нули. Показания сглаживают скользящим средним. Нужно, чтобы после сглаживания вспышка поднималась выше 5, а блик оставался ниже 1. Какое окно подходит: 3, 5 или 7? В ответ введи его ширину.

Дано: вспышка – три значения 9; блик – одно значение 3; вокруг нули; окна 3, 5, 7.

Условие: вспышка после сглаживания больше 5, блик меньше 1.

Найти: подходящее окно.

Решение

Шаг 1. Самое большое значение вспышки после сглаживания. Выше всего среднее выйдет тогда, когда в окно попадут все три девятки. Их сумма $9 + 9 + 9 = 27$, остальные числа в окне – нули. Делим на ширину окна:

- окно 3: $27:3 = 9$;
- окно 5: $27:5 = 5,4$;
- окно 7: $27:7 \approx 3,9$.

Шаг 2. Блик после сглаживания. В любом окне, где есть блик, сумма равна 3, остальное – нули:

- окно 3: $3:3 = 1$;
- окно 5: $3:5 = 0,6$;
- окно 7: $3:7 \approx 0,43$.

Шаг 3. Сравниваем с условием.

Таблица 3.2

Окно	Вспышка	Блик	Годится
3	9	1	нет
5	5,4	0,6	да
7	3,9	0,43	нет

Окно 3 не годится из-за блика: он равен 1, а нужно меньше 1. Окно 7 не годится из-за вспышки: 3,9 не выше 5. Остаётся окно 5.

Картина та же, что в разделе 3.4: узкое окно пропускает блик, широкое съедает саму вспышку, и хорошее окно оказывается посередине.

⚠ Типичная ошибка. Делить на ширину окна одну девятку: $9:5 = 1,8$, $9:3 = 3$ – и решить, что вспышку съест любое окно. Но в окно попадают все три девятки подряд, их сумма 27. Вторая ловушка – слово «ниже»: блик 1 при окне 3 на границе, а граница условию не подходит.

Ответ: 5.

Своя задача по программе сферы, модуль 3

Реши сам

3.7. Вспышка длится 4 измерения подряд со значением 6, блик – одно измерение со значением 4, вокруг нули. Нужно, чтобы после сглаживания вспышка была выше 3, а блик ниже 0,6. Какое окно подходит: 3, 5, 7 или 9?

3.8. Вспышка – 2 измерения подряд со значением 12, вокруг нули. Ширину окна можно выбрать любую целую, от 2 до 10. Какое самое широкое окно ещё оставит вспышку после сглаживания не ниже 4?

Задача 3.5 Сообщение в огибающей

несущая, огибающая, выпрямление, скользящее среднее

Передатчик отправляет биты: 1 – сильные колебания несущей, 0 – слабые. а) Частота несущей 4000 Гц, один бит длится 0,01 с. Сколько колебаний несущей укладывается в один бит? б) Приёмник записал по 8 чисел на каждый бит:

Таблица 3.3

Бит	Записанные значения
первый	0, 6, 0, -6, 0, 6, 0, -6
второй	0, 2, 0, -2, 0, 2, 0, -2
третий	0, 6, 0, -6, 0, 6, 0, -6

Выдели огибающую: выпрями сигнал и сгладь его скользящим средним с окном 4. Бит равен 1, если огибающая внутри бита больше 2, и 0 – если меньше. Какие биты переданы? В ответ введи через пробел число из а) и три бита подряд.

Дано: $f = 4000$ Гц; $t = 0,01$ с; записанные значения в таблице; окно 4; порог 2.

Найти: число колебаний на бит N ; переданные биты.

Решение

Шаг 1. Колебания на бит. За секунду несущая делает 4000 колебаний, за 0,01 с – в сто раз меньше: $N = f \cdot t = 4000 \cdot 0,01 = 40$.

Шаг 2. Выпрямляем. Все –6 и –2 превращаются в 6 и 2:

- первый бит: 0, 6, 0, 6, 0, 6, 0, 6;
- второй бит: 0, 2, 0, 2, 0, 2, 0, 2;
- третий бит: 0, 6, 0, 6, 0, 6, 0, 6.

Шаг 3. Сглаживаем окном 4. Внутри первого бита любые четыре соседних числа – это два нуля и две шестёрки: $(0 + 6 + 0 + 6) : 4 = 12 : 4 = 3$. Внутри второго бита: $(0 + 2 + 0 + 2) : 4 = 1$. Внутри третьего снова 3. Окна на стыке двух битов захватывают кусочки обоих, например 0, 6, 0, 2: $8 : 4 = 2$. Это переходы, по ним бит не читают – смотрим в середину бита.

Шаг 4. Читаем биты. 3 больше 2 – это 1. 1 меньше 2 – это 0. Снова 3 – это 1. Переданы биты 101.

Огибающая вышла 3 при амплитуде 6 – та же история, что с примером в разделе 3.5. Для чтения битов это не помеха: 3 и 1 спутать трудно. А 40 колебаний несущей на один бит – ещё одно напоминание, насколько несущая быстрее огибающей.

⚠ Типичная ошибка. Сглаживать без выпрямления. Тогда в каждом окне $(0 + 6 + 0 - 6) : 4 = 0$, у всех битов огибающая 0, и сообщение пропало. В пункте а) бывает другая промашка – разделить вместо умножения: $4000 : 0,01 = 400\,000$. Частота – это колебания за секунду, а бит длится меньше секунды, значит, колебаний в нём меньше 4000.

Ответ: 40 101.

Своя задача по программе сферы, модуль 3

Реши сам

3.9. Частота несущей 1500 Гц, огибающая – медленная волна частотой 30 Гц. Сколько колебаний несущей укладывается в один период огибающей?

3.10. Приёмник записал два бита по 8 чисел: первый – 0, 5, 0, –5, 0, 5, 0, –5; второй – 0, 1, 0, –1, 0, 1, 0, –1. Выпрями сигнал и сгладь окном 4. Чему равна огибающая внутри первого и внутри второго бита? Какие биты переданы, если порог 1,5? Введи через пробел два значения огибающей и два бита подряд.

Задача 3.6 Аномалия в сообщении метеостанции

аномалия, выброс, отрицательные числа, среднее

Школьная метеостанция в Бердске каждые 10 минут передаёт по радио температуру воздуха. За два часа пришло 12 значений, в градусах Цельсия:

-14, -13, -13, -12, -12, 19, -11, -11, -10, -10, -9, -9.

Одно значение искажилось в канале связи. Аномалией считаем значение, которое отличается от **каждого** из двух соседей больше чем на 5 °С. а) Найди номер аномального значения (считая с 1). б) Замени его средним двух соседей. в) На сколько градусов искажённое значение завишило среднюю температуру за два часа? Округли до десятых. В ответ введи три числа через пробел.

Дано: 12 значений температуры; порог 5 °С.

Найти: номер выброса; исправленное значение; сдвиг среднего.

Решение

Шаг 1. Ищем выброс. Идём по ряду и сравниваем каждое число с соседями. Почти везде соседи отличаются на 0 или 1 градус. Выбивается шестое значение, 19. Насколько оно дальше от пятого, -12? Посмотри на уличный градусник: от -12 до нуля столбик поднимается на 12 делений, от нуля до 19 – ещё на 19. Всего $12 + 19 = 31$ градус. От седьмого значения, -11, так же выходит $11 + 19 = 30$ градусов. Оба раза больше 5 – это аномалия. А пятое значение, -12, от 19 тоже сильно отличается, но от своего левого соседа, -12, – на 0. Отличие только от одного соседа по условию аномалией не считается.



Рисунок 3.13 – Расстояние от -12 до 19 на градуснике: сначала до нуля, потом от нуля

Шаг 2. Исправляем. Нужно среднее двух соседей, -12 и -11. На градуснике между ними одно деление, и середина приходится на -11,5 °С. То же по правилу: складывая два отрицательных числа, складываем их без минусов и ставим минус, $(-12) + (-11) = -23$, а $-23:2 = -11,5$.

Шаг 3. Сдвиг среднего. Среднее двенадцати значений – их сумма, делённая на 12. Одно значение ошибочно стоит 19 вместо -11,5, то есть завышено на $11,5 + 19 = 30,5$ градуса (снова по градуснику: до нуля и от нуля). Значит, и сумма завышена на 30,5 градуса. Эти лишние 30,5 делятся на 12 значений: $30,5:12 \approx 2,54$. После округления до десятых – 2,5 °С.

Кто уже умеет делить отрицательные числа, может проверить длинным путём. Среднее с ошибкой $-105:12 = -8,75$ °С, без ошибки $-135,5:12 \approx -11,29$ °С, разница те же 2,54.

Исправленное -11,5 °С хорошо встаёт в ряд: за два часа воздух медленно потеплел с -14 до -9, и в середине этого пути должно быть около -11. А плюс 19 в сибирскую зиму на десять минут посреди мороза не бывает.

! Типичная ошибка. Отметить аномалией ещё и пятое или седьмое значение: «оно же сильно отличается от 19». Условие требует отличия от каждого соседа, и такое число одно. Вторая частая ошибка – потерять минус: $(-12 - 11) : 2 = 11,5$ вместо $-11,5$. Температура вокруг -12 , а исправленное вышло плюсовым – сразу видно, что где-то потерялся знак.

Ответ: 6 $-11,5$ 2,5.

По мотивам задачи заключительного этапа НТО 2025/26, профиль «Технологии беспроводной связи», «УНКС: нестабильный прием»

Реши сам

3.11. Спектр вибрации исправного насоса: 25 Гц (амплитуда 5), 50 Гц (амплитуда 2), 75 Гц (амплитуда 1). Через месяц спектр такой: 25 Гц (5), 50 Гц (2), 75 Гц (1), 110 Гц (3). Какая частота – аномалия? Введи её в герцах.

3.12. Маяк на аэростате без конца повторяет одно и то же сообщение из пяти цифр. Приёмник записал 20 цифр: 2 7 7 4 9 2 7 7 4 9 2 7 7 1 9 2 7 7 4 9. Одна цифра искажена. Найди её номер, считая с 1.

Задача 3.7* Окно, которое гасит гул

скользящее среднее, период, наводка, частота измерений

Датчик на учебном стенде измеряет напряжение 200 раз в секунду. В проводах есть наводка от электросети частотой 50 Гц: к полезному сигналу прибавляется повторяющийся узор +3, 0, -3, 0. Полезный сигнал меняется медленно, и на коротком отрезке он равен 20. Пришли значения:

23, 20, 17, 20, 23, 20, 17, 20.

а) Сколько измерений приходится на один период наводки? б) Сгладь ряд окном 3 и окном 4. Какое из них убрало наводку полностью? в) Какие окна из списка 3, 4, 5, 6, 8 гасят наводку 50 Гц полностью? г) Какое самое узкое окно полностью погасит наводку частотой 40 Гц? В ответ введи все числа через пробел: ответ а), окна из пункта в) по возрастанию, ответ г).

Дано: 200 измерений в секунду; наводка 50 Гц с узором +3, 0, -3, 0; значения в условии.

Найти: измерений на период; какие окна гасят наводку 50 Гц; окно для 40 Гц.

Решение

Шаг 1. Период наводки в измерениях. Период – это время одного колебания:

$T = 1 : 50 = 0,02$ с. За 0,02 с датчик успевает сделать $200 \cdot 0,02 = 4$ измерения. То же в одно действие: $200 : 50 = 4$. В узоре +3, 0, -3, 0 как раз четыре числа.

Шаг 2. Окно 3. $(23 + 20 + 17) : 3 = 20$; $(20 + 17 + 20) : 3 = 19$; $(17 + 20 + 23) : 3 = 20$; $(20 + 23 + 20) : 3 = 21$; дальше снова 20 и 19. Вышло 20, 19, 20, 21, 20, 19 – сигнал всё ещё качается на единицу вверх и вниз.

Шаг 3. Окно 4. Любые четыре соседних числа – это 23, 20, 17 и 20 в каком-то порядке. Их сумма 80, среднее $80:4 = 20$. Вышло 20, 20, 20, 20, 20 – наводки не осталось, получился чистый полезный сигнал.

Шаг 4. Почему окно 4 сработало. В окно 4 помещается один целый период наводки: +3 и –3 гасят друг друга, нули ничего не добавляют. Окно 8 вмещает два целых периода – тоже гасит. В окна 3, 5 и 6 попадает кусок периода, и остаток вылезает: например, окно 5 с кусочком +3, 0, –3, 0, +3 даёт $3:5 = 0,6$ вместо нуля. Правило: окно полностью гасит наводку, если в нём помещается целое число её периодов.

Шаг 5. Наводка 40 Гц. На один её период приходится $200:40 = 5$ измерений. Самое узкое окно, в котором целый период, – 5.

Этим приёмом пользуются и старшеклассники на финале НТО по беспроводной связи: подбирают окно так, чтобы оно гасило частоту чужого передатчика и оставляло свою.

 **Типичная ошибка.** Думать, что чем шире окно, тем чище: окно 6 шире окна 4, а наводку не убирает, потому что полтора периода не гасят друг друга. Ещё путают деление: $50:200 = 0,25$. Четверть измерения на период? Так не бывает: измерений на период больше одного, значит, делим большее число на меньшее.

Ответ: 4 4 8 5.

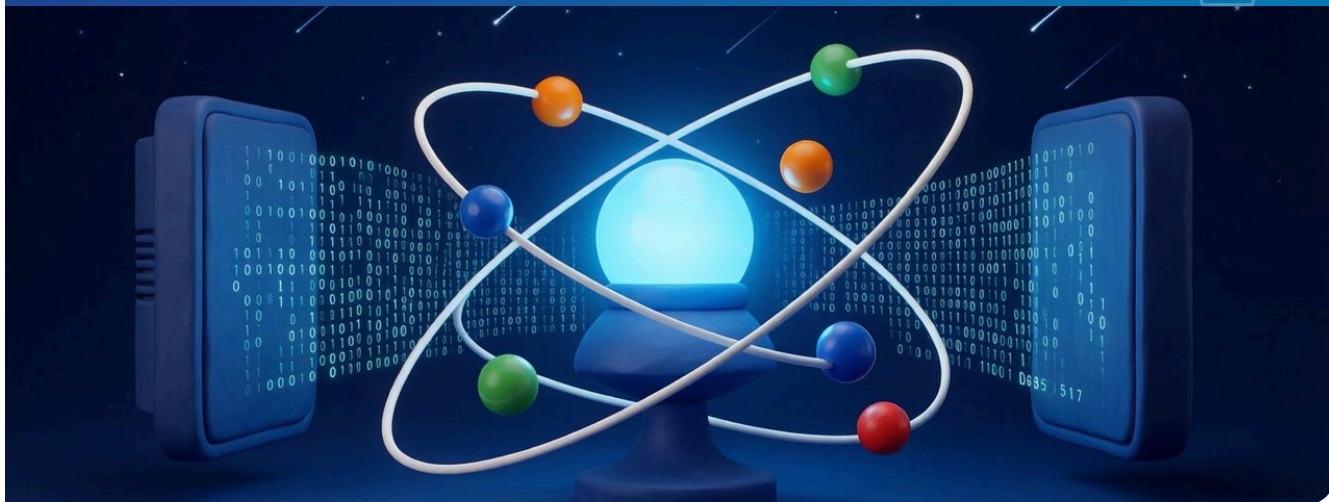
По мотивам задачи заключительного этапа НТО 2025/26, профиль «Технологии беспроводной связи», «Полигон-А: деконволюция»

Реши сам

3.13.* Датчик измеряет 1000 раз в секунду. Мешают две наводки: 250 Гц и 200 Гц. Какое самое узкое окно скользящего среднего полностью погасит обе сразу?

3.14.* Датчик измеряет 120 раз в секунду, сигнал сглаживают окном 12. Какие наводки это окно гасит полностью: 1) 10 Гц; 2) 20 Гц; 3) 25 Гц; 4) 30 Гц; 5) 40 Гц? Введи номера по возрастанию, без пробелов.

Кодирование и системы счисления



Компьютер, КиберМишка и пульт от телевизора различают только два знака – «есть сигнал» и «нет сигнала», 1 и 0. Из этих двух знаков собирают и числа, и буквы, и команды роботу. Здесь ты научишься переводить числа в двоичный код и обратно, считать, сколько бит займёт сообщение и сколько секунд оно пробудет в канале, и ловить ошибки, которые помеха вносит по дороге. На отборе такие задачи короткие, а ответ – число или строка из цифр: перепутал одну цифру – и весь ответ не засчитан.

4.1. Код, бит и байт

НА ПАЛЬЦАХ

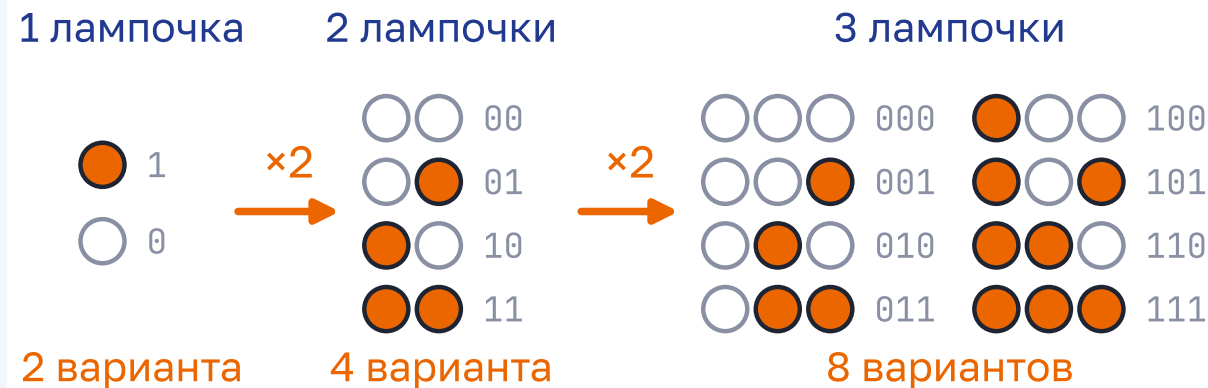


Рисунок 4.1 – Одна лампочка, две, три – и все положения, в которых они могут гореть

Одна лампочка скажет только два слова: горит или не горит. Добавил вторую – вариантов стало четыре, третью – восемь. Каждая новая лампочка удваивает счёт.

Код – договорённость о том, какой сигнал что значит. Бабушка на даче вешает на калитку красную тряпку, когда обед готов, – это код, пусть и из одного знака. **Кодирование** – перевод сообщения в сигналы по договорённости, **декодирование** – обратный перевод на стороне приёмника. Знаки, из которых собирают код, называют **алфавитом кода**: у азбуки Морзе это точка и тире, у светофора – три цвета, у компьютера – 0 и 1. Цепочка знаков, которая обозначает одну букву или одну команду, – **кодовое слово**.

Почему компьютеру хватает двух знаков? «Есть ток» и «нет тока» легко различить даже при сильной помехе. Будь знаков десять, прибору пришлось бы отличать друг от друга десять уровней напряжения, и малейшая дрожь в проводе путала бы тройку с четвёркой. Помнишь лестницу из главы 2? У цифрового сигнала компьютера в ней всего две ступеньки.

Один знак двоичного кода – 0 или 1 – называется **бит** (от английского binary digit, «двоичная цифра»). Бит – самая маленькая порция информации, один ответ «да» или «нет». Восемь бит подряд – **байт**.

Сколько разных сообщений помещается в n бит? В одном бите – два: 0 и 1. В двух – четыре: 00, 01, 10, 11. К каждому старому варианту можно приписать справа и 0, и 1, поэтому каждый новый бит удваивает число вариантов. Словами: **число разных кодовых слов равно двойке, умноженной на себя столько раз, сколько бит**. Буквами:

$$N = 2^n,$$

где n – число бит, N – сколько разных кодовых слов можно из них составить. Степени двойки полезно помнить до восьмой: $2^1 = 2$, $2^2 = 4$, $2^3 = 8$, $2^4 = 16$, $2^5 = 32$, $2^6 = 64$, $2^7 = 128$, $2^8 = 256$. Значит, в одном байте 256 вариантов – числа от 0 до 255. Не до 256: первым вариантом идёт ноль.

В задачах вопрос часто задают наоборот: сообщений M штук, сколько бит нужно на каждое? Ищем самую маленькую степень двойки, которая не меньше M . В английском алфавите 26

букв: $2^4 = 16$ мало, $2^5 = 32$ хватает – нужно 5 бит, и 6 кодов останутся свободными. В русском алфавите 33 буквы: $2^5 = 32$, одной букве кода не хватит, поэтому берут 6 бит. Число бит всегда округляют вверх. Код, в котором все кодовые слова одной длины, называют **равномерным**.

Если на каждый знак уходит i бит, а знаков в сообщении K , всё сообщение займёт

$$I = K \cdot i \text{ бит.}$$

Слово из 12 букв кодом по 5 бит – это $12 \cdot 5 = 60$ бит.

Так компьютер хранит и текст. Все буквы, цифры и знаки выписаны в **таблицу символов**, у каждого там свой номер, а в память кладётся номер. В самой известной таблице, ASCII («аски»), латинская А стоит под номером 65, В – под номером 66, и дальше подряд до Z – 90. Цифра «0» в этой таблице – знак номер 48. Так что знак «0» в тексте и число 0 в памяти – для компьютера разные вещи.

Если знаков в алфавите кода больше двух, считают так же. Фонарик со стёклами трёх цветов и две вспышки подряд дают $3 \cdot 3 = 9$ разных сигналов, три вспышки – $3 \cdot 3 \cdot 3 = 27$.

Запомни. Бит – одна двоичная цифра, байт – 8 бит. В n бит помещается 2^n разных кодов, в байт – 256, это числа от 0 до 255. Сколько бит нужно на M сообщений: найди первую степень двойки, которая не меньше M . Вниз не округляют никогда.

4.2. Двоичная система счисления

НА ПАЛЬЦАХ

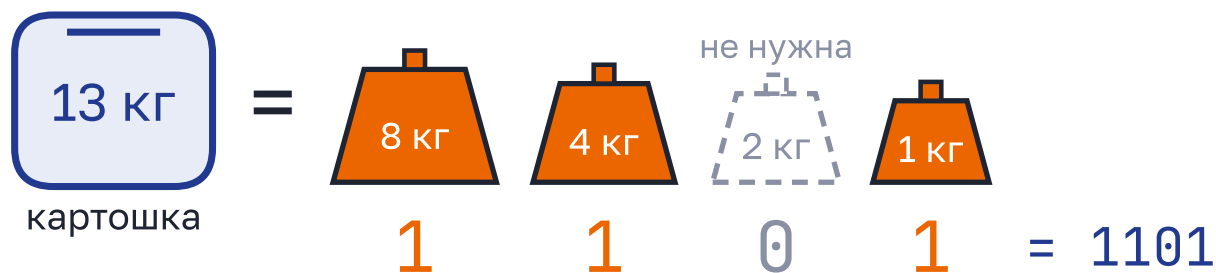


Рисунок 4.2 – 13 кг картошки уравнивают гирями 8, 4, 2 и 1 кг – каждая гиря есть только одна

Гиря на весах – пишем 1, гиря осталась на полу – 0. На весах 8, 4 и 1 кг, двухкилограммовая не понадобилась. Число 13 в двоичной системе – 1101.

Наши обычные числа записаны в **десятичной системе**: цифр десять, от 0 до 9, и у каждого места в числе свой вес. В числе 305 тройка стоит на месте сотен и значит 300, ноль – на месте десятков, пятёрка – на месте единиц: $305 = 3 \cdot 100 + 0 \cdot 10 + 5 \cdot 1$. Веса мест – 1, 10, 100, 1000, каждый следующий в 10 раз больше.

В **двоичной системе** цифр две, 0 и 1, а вес каждого следующего места вдвое больше предыдущего: 1, 2, 4, 8, 16, 32, 64, 128. Это те самые гири с картинки. Чтобы не спутать системы, осно-

вание пишут маленькой цифрой внизу: 1101_2 – двоичное число, 1101_{10} – обычное «тысяча сто один».

Из двоичной в десятичную. Подпиши над каждой цифрой её вес – **справа налево**, начиная с 1. Сложи веса там, где стоят единицы. У 1101_2 справа налево веса 1, 2, 4, 8, единицы стоят под 8, 4 и 1: $8 + 4 + 1 = 13$.

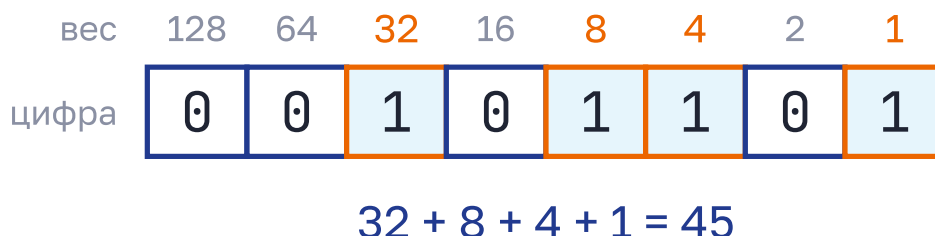


Рисунок 4.3 – Байт 00101101_2 : над цифрами – веса мест, складываем только те, под которыми стоит единица

Из десятичной в двоичную – гирями. Берём самую тяжёлую гирю, которая не больше числа, и вычитаем её. С остатком делаем то же самое. Число 45: гиря 32 подходит, остаётся 13; гиря 16 больше 13 – пропускаем; 8 подходит, остаётся 5; 4 подходит, остаётся 1; 2 пропускаем; 1 подходит, остаётся 0. Взяли гири 32, 8, 4 и 1. На их местах ставим единицы, на остальных нули: $45 = 101101_2$.

Из десятичной в двоичную – делением. Дели число на 2 и записывай остатки, пока частное не станет нулём. Потом читай остатки **снизу вверх**: последний остаток – самая левая цифра.

Таблица 4.1

Что делим на 2	Частное	Остаток
45	22	1
22	11	0
11	5	1
5	2	1
2	1	0
1	0	1

Снизу вверх: 1, 0, 1, 1, 0, 1 – снова 101101_2 . Почему это работает? Остаток от деления на 2 говорит, чётное число или нечётное, то есть нужна ли гиря в 1 кг. Разделили пополам – и тем же вопросом решаем про гирю в 2 кг, потом в 4 кг. Отсюда же полезное следствие: **чётное число в двоичной записи кончается нулём, нечётное – единицей**.

В байте всегда восемь мест. Если число короче, слева дописывают нули: 45 в байте – 00101101 . Значение от этого не меняется, как «007» – это всё те же 7. Самое большое число в байте – $11111111_2 = 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255$. Прибавь единицу – получится $100000000_2 = 256$, девять цифр, в байт уже не влезает. Кстати, если на пальцах считать «поднят – 1, загнут – 0», двумя руками можно досчитать до $2^{10} - 1 = 1023$.

Проверить себя поможет Python (подойдёт любой онлайн-интерпретатор) или калькулятор Windows в режиме «Программист».

```
print(int("00101101", 2)) # 45 - читаем строку как двоичное число
print(format(45, "08b"))  # 00101101 - пишем 45 двоичными цифрами, 8 знаков
print(bin(45))            # 0b101101 - приписку 0b в ответ не переписывай
```

Первая строка: `int` превращает строку в число, а двойка после запятой говорит, что цифры в строке двоичные. Вторая: `format` записывает число 45 по шаблону `"08b"` – `b` значит «двоичными цифрами», `08` – «восемь знаков, недостающие заполни нулями». Всё, что после `#`, – комментарий, Python его не читает.

Запомни. Веса двоичных мест справа налево: 1, 2, 4, 8, 16, 32, 64, 128. Из двоичной – сложи веса под единицами. В двоичную – набирай число гирями, начиная с самой тяжёлой, или дели на 2 и читай остатки снизу вверх. Байт – это всегда 8 цифр: недостающие нули пиши слева.

Совет. Любой перевод проверяй обратным ходом. Получил 1001011_2 – сложи веса под единицами и убедись, что снова вышло твоё число. Это полминуты, а ошибку в одной цифре ты поймашь до того, как введёшь ответ.

4.3. Шестнадцатеричная система

НА ПАЛЬЦАХ

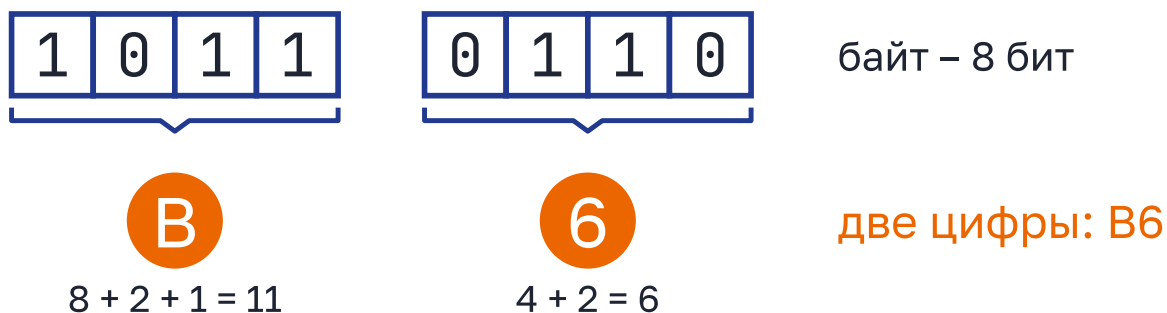


Рисунок 4.4 – Восемь нулей и единиц режут на две четвёрки, и каждая четвёрка становится одной цифрой

Номер телефона тоже диктуют кусочками – так меньше шансов сбиться. Байт 10110110 режут на 1011 и 0110, и получаются две цифры: B и 6.

В **шестнадцатеричной системе** шестнадцать цифр. Первые десять – привычные 0–9. Для значений от 10 до 15 взяли латинские буквы: A = 10, B = 11, C = 12, D = 13, E = 14, F = 15. Вес мест растёт в 16 раз: единицы, потом «шестнадцатки», потом 256. Для чисел до 255 хватает двух цифр.

Зачем она нужна, если есть двоичная? Чтобы людям было удобно читать. $16 = 2^4$, поэтому каждая шестнадцатеричная цифра заменяет четыре бита, а байт записывается двумя цифр-

рами – от 00 до FF. Длинную цепочку 10110110 легко переписать с ошибкой, а B6 – трудно. Поэтому программисты пишут байты в шестнадцатеричной записи.

Четвёрки бит и цифры стоит выучить или держать под рукой:

Таблица 4.2

Цифра	Четыре бита	Цифра	Четыре бита
0	0000	8	1000
1	0001	9	1001
2	0010	A (10)	1010
3	0011	B (11)	1011
4	0100	C (12)	1100
5	0101	D (13)	1101
6	0110	E (14)	1110
7	0111	F (15)	1111

Четвёрку легко восстановить и без таблицы: это двоичная запись числа от 0 до 15 с весами 8, 4, 2, 1. D = 13 = 8 + 4 + 1, значит, D – это 1101.

- **Из шестнадцатеричной в двоичную:** каждую цифру замени её четвёркой.
 $B6_{16} = 1011\ 0110_2$.
- **Из двоичной в шестнадцатеричную:** режь на четвёрки **справа налево**, слева допиши нули до полной четвёрки. $101101_2 = 0010\ 1101_2 = 2D_{16}$.
- **Из шестнадцатеричной в десятичную:** старшую (левую) цифру умножь на 16 и прибавь младшую. $B6_{16} = 11 \cdot 16 + 6 = 176 + 6 = 182$.
- **Из десятичной в шестнадцатеричную:** раздели на 16 с остатком. Частное – старшая цифра, остаток – младшая. $200 = 12 \cdot 16 + 8$, то есть $200 = C8_{16}$.

Где это встретится в жизни? В цвете на экране. Цвет каждой точки задают три числа от 0 до 255: сколько в нём красного (R), зелёного (G) и синего (B). Каждое число – байт, весь цвет – три байта, и в программах для рисования и на сайтах его пишут шестью шестнадцатеричными цифрами после решётки: #FF0000 – чистый красный, #FFFFFF – белый, #000000 – чёрный. Разных цветов так можно задать $2^{24} = 16\ 777\ 216$.

В Python: `int("B6", 16)` даёт 182, а `format(182, "02X")` – строку B6 (две цифры, большие буквы).

Запомни. A = 10, B = 11, C = 12, D = 13, E = 14, F = 15. Одна шестнадцатеричная цифра – четыре бита, байт – две цифры. Двоичное число режь на четвёрки справа. В десятичную: старшая цифра · 16 + младшая; обратно – дели на 16 с остатком.

4.4. Скорость передачи

НА ПАЛЬЦАХ

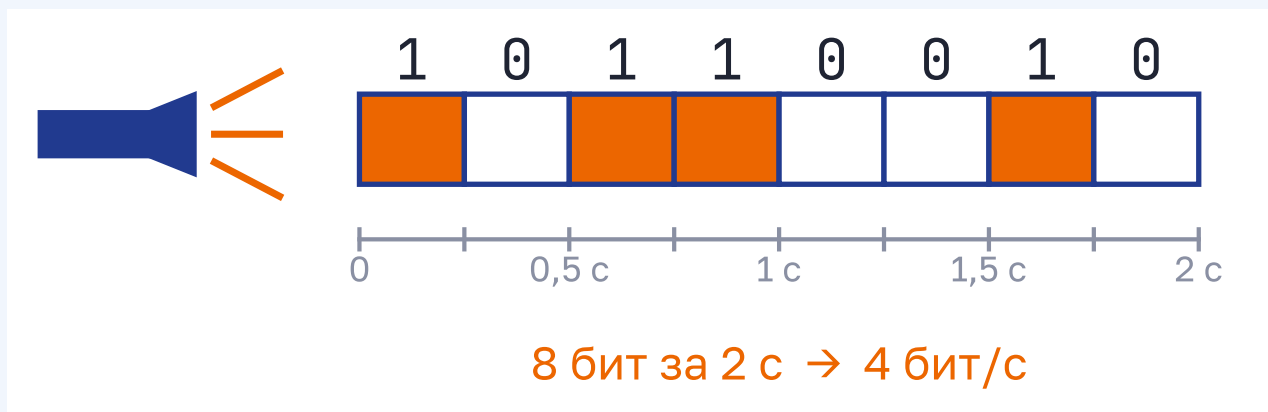


Рисунок 4.5 – Фонарик передаёт байт: на каждый бит отведена своя четверть секунды

Каждую четверть секунды фонарик либо горит – это 1, либо нет – это 0. За 2 секунды успевают пройти 8 бит, по 4 бита в каждую секунду.

По каналу связи биты идут друг за другом, по очереди. **Скорость передачи** – сколько бит проходит по каналу за одну секунду. Единица – бит в секунду, бит/с. Тысяча бит в секунду – 1 кбит/с.

Время передачи считают как время в пути. Словами: **время равно объёму сообщения, делённому на скорость**. Буквами:

$$t = \frac{I}{v},$$

где I – объём сообщения в битах, v – скорость в бит/с, t – время в секундах. Биты делим на биты в секунду – остаются секунды. Слово «ПРИВЕТ» – 6 букв по 8 бит, это 48 бит. При скорости 12 бит/с оно уйдёт за $48 : 12 = 4$ с.

У каждого бита есть **длительность** – сколько времени он занимает в канале, одна клеточка на рисунке. Её обозначают греческой буквой τ («тау»). Если за секунду проходит v бит, на один бит приходится

$$\tau = \frac{1}{v} \text{ с.}$$

У фонарика с рисунка $v = 4$ бит/с и $\tau = 1 : 4 = 0,25$ с. Работает и в обратную сторону: бит длится 10 мс, то есть 0,01 с, – значит, скорость $1 : 0,01 = 100$ бит/с.

Самая коварная ловушка – биты и байты. Скорость почти всегда дают в битах в секунду, а размер сообщения – в символах или байтах. Прежде чем делить, переведи всё в биты: байты умножь на 8. И следи за приставками: 1 кбит/с = 1000 бит/с, а 1 Кбайт в школьной информатике – 1024 байта, если в условии не сказано иначе.

Часто к каждому байту передатчик добавляет свои **служебные биты** – например, бит чётности из следующего раздела. Их тоже надо передать, и они тоже занимают время. Байт с одним служебным битом – это 9 бит в канале, и сообщение идёт на девятую часть дольше.

Запомни. $t = I : v$: биты делим на бит/с, получаем секунды. Один бит длится $1 : v$ секунды. Байты сначала умножь на 8, килобиты в секунду – на 1000. Служебные биты тоже едут по каналу.

4.5. Как поймать ошибку: чётность и контрольная сумма

НА ПАЛЬЦАХ

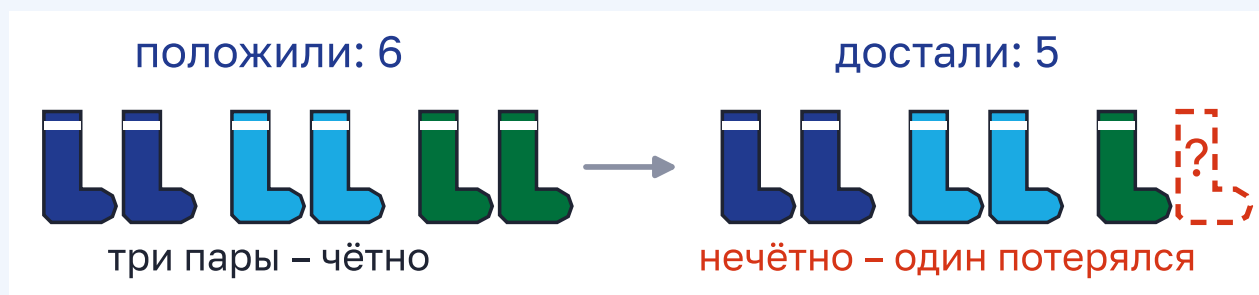


Рисунок 4.6 – Носки из стиральной машины

В машинку положили три пары носков – чётное число. Достали пять: число нечётное, значит, один застрял в барабане. Какой из шести, по одному пересчёту не скажешь.

В любом канале бывают **помехи**: треск в эфире, солнечный зайчик на ИК-приёмнике, хлопок двери рядом с микрофоном. Иногда помеха переворачивает бит: отправили 0, пришла 1. Сам приёмник этого не видит – испорченная единица выглядит так же, как настоящая. Поэтому к сообщению добавляют несколько лишних, **проверочных** бит, по которым приёмник замечает неладное.

Бит чётности. Передатчик считает единицы в байте и дописывает в конец ещё один бит – такой, чтобы единиц стало чётное число. В `01001011` четыре единицы, уже чётно, дописываем 0: `010010110`. В `11100000` три единицы, дописываем 1: `111000001`. Приёмник считает единицы во всех девяти битах. Чётно – байт принят. Нечётно – где-то перевернулся бит, и байт просят прислать снова. Если испортится сам бит чётности, это тоже заметно: он ведь тоже входит в подсчёт.

Но **какой** бит испорчен, бит чётности не скажет – как и пересчёт носков. А две ошибки сразу он вообще пропустит: две перевёрнутые цифры оставляют число единиц чётным. Помехи бывают редко, и две ошибки в одном байте случаются куда реже одной, поэтому бит чётности всё равно полезен.

```
bajt = "001100001"      # 9 бит: 8 бит данных и бит чётности
print(bajt.count("1"))   # 3 – нечётно, значит, байт испорчен
```

`count("1")` считает, сколько раз в строке встречается единица. Дальше остаётся посмотреть, чётное ли вышло число.

Контрольная сумма. Когда передают числа, в конце можно послать ещё одно – их сумму. Приёмник сам складывает полученные числа и сравнивает результат с присланной суммой. Отправили 12, 7, 30 и сумму 49. Пришло 12, 7, 38 и 49: $12 + 7 + 38 = 57$, а в конце стоит 49. Значит, по дороге что-то испортилось.

Чтобы контрольная сумма помещалась в один байт, от суммы часто оставляют только **остаток от деления на 256**: сумма 300 превращается в $300 - 256 = 44$. Как считать контрольную сумму, каждый раз написано в условии задачи, и это правило нужно выполнить до последней буквы. Слабое место есть и у неё: если два числа поменялись местами, сумма не изменится, и такую ошибку проверка пропустит.

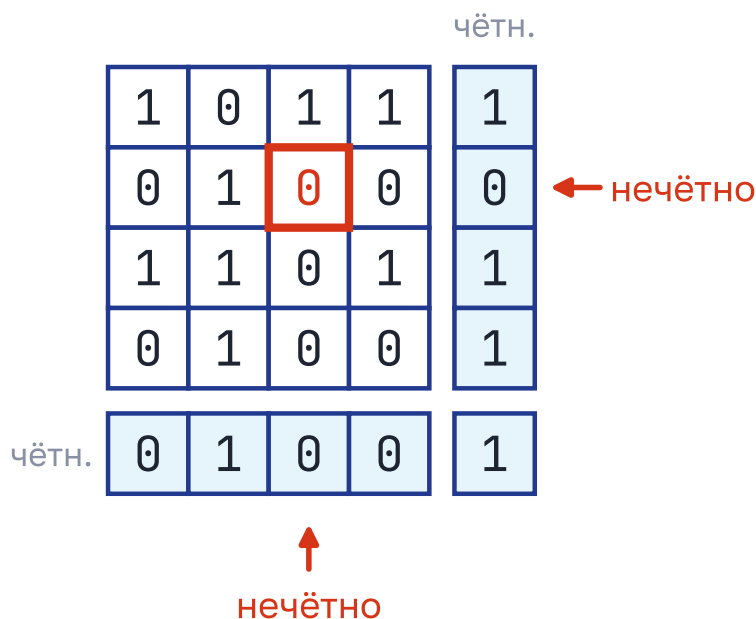


Рисунок 4.7 – Чётность по строкам и столбцам. Нечётными стали одна строка и один столбец – ошибка на их пересечении

Для любопытных: чётность по строкам и столбцам. Сложим биты в таблицу по четыре в строке и допишем бит чётности к каждой строке и к каждому столбцу. Теперь одна ошибка выдаёт себя дважды: нечётной становится одна строка и один столбец. На их пересечении и сидит испорченный бит. Его можно не только найти, но и перевернуть обратно, не переспрашивая передатчик.

Запомни. Бит чётности делает число единиц чётным. На приёме нечётно – ошибка есть, но где она, неизвестно; двух ошибок сразу он не замечает. Контрольная сумма – сумма всех чисел, её правило (например, остаток от деления на 256) всегда даётся в условии.

4.6. Азбука Морзе

НА ПАЛЬЦАХ

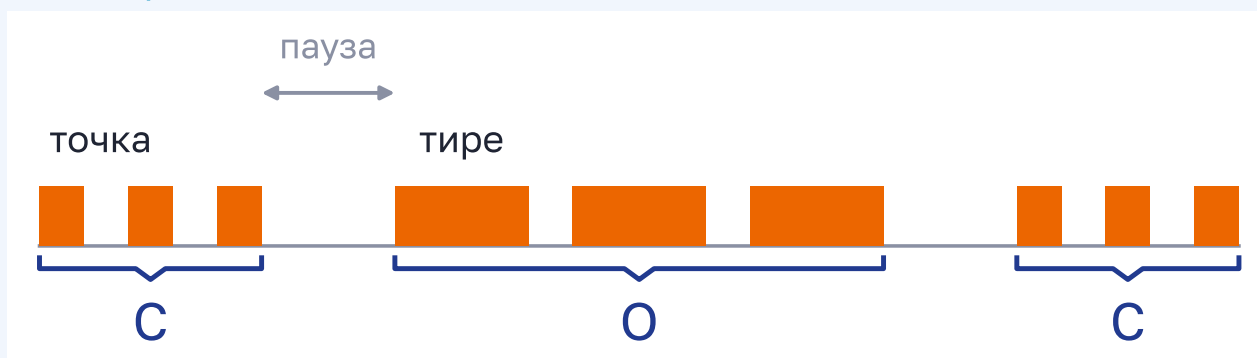


Рисунок 4.8 – Сигнал бедствия SOS фонариком: три коротких вспышки, три длинных, три коротких

Короткая вспышка – точка, длинная – тире. Паузы тоже часть кода: без них не понять, где кончилась одна буква и началась следующая.

Азбуку Морзе придумали почти двести лет назад для телеграфа: по проводу посылали короткие и длинные толчки тока, а на другом конце их записывали на бумажную ленту или принимали на слух. Потом той же азбукой переговаривались прожекторами с корабля на корабль и по радио.

Алфавит кода Морзе – точка и тире. Буква Е – одна точка, Т – одно тире, А – .- , О – --- . Коды всех букв и цифр – в справочнике в конце книги. Этот код **неравномерный**: у одних букв кодовое слово короче, у других длиннее. Самые короткие коды достались буквам, которые чаще встречаются в тексте, – так сообщение выходит короче.

Сколько кодов Морзе можно составить из одного–четырёх знаков? Из одного знака – 2, из двух – $2^2 = 4$, из трёх – 8, из четырёх – 16, всего $2 + 4 + 8 + 16 = 30$. Для 26 латинских букв хватает. В русском алфавите букв больше, поэтому Ъ и Э получили коды из пяти знаков, из пяти знаков состоят и коды цифр.

Длительности в Морзе считают в «точках»:

- точка – 1 единица времени, тире – 3;
- пауза между знаками внутри буквы – 1;
- пауза между буквами – 3, между словами – 7.

Посчитаем слово «ДА». Д – -.. : тире 3, пауза 1, точка 1, пауза 1, точка 1, всего 7 единиц. Пауза между буквами – 3. А – .- : 1 + 1 + 3 = 5. Всё слово: $7 + 3 + 5 = 15$ единиц. Если точка длится 0,2 с, «ДА» займёт $15 \cdot 0,2 = 3$ с.

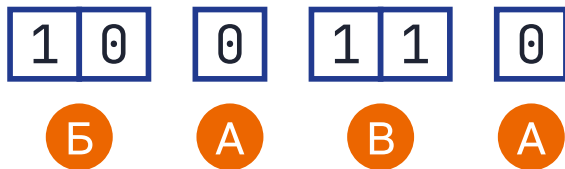
Зачем столько внимания паузам? Приходит ... – это С? Или И (..) и Е (.)? Или три буквы Е? Без пауз не разобрать. В Морзе пауза – полноправный знак кода, хоть её и не рисуют.

Запомни. Морзе: точка 1, тире 3, паузы 1 – 3 – 7 (внутри буквы, между буквами, между словами). Чтобы узнать длительность слова, сложи все тоны и все паузы между ними.

4.7. Свой код без разделителей

НА ПАЛЬЦАХ

A = 0, Б = 10, В = 11



режется одним способом

Б = 1, В = 11



Рисунок 4.9 – Одна и та же цепочка при хорошем и при сломанном словаре

В хорошем словаре цепочку можно разрезать только одним способом. Сделай Б коротким, 1, – и 11 читается и как В, и как ББ. С телефоном было бы так же: будь у соседа номер 11, до службы спасения 112 никто бы не дозвонился – звонок уходил бы соседу сразу после второй единицы.

Свой код придумать может каждый – нужно только договориться с тем, кто будет принимать. Проще всего равномерный код из раздела 4.1: приёмник режет сигнал на кусочки по n бит, и разделители не нужны. Для четырёх команд робота хватит двух бит: ВПЕРЁД – 00, НАЗАД – 01, НАЛЕВО – 10, НАПРАВО – 11. Цепочка 001011 режется на 00, 10, 11: вперёд, налево, направо.

Неравномерный код выходит короче, если частым сообщениям дать короткие слова. Но приёмнику надо как-то понять, где кончается одно слово и начинается следующее. Помогает правило: **ни одно кодовое слово не должно быть началом другого**. Возьмём код А – 0, Б – 10, В – 11. Правило выполнено: 0 не начинает ни 10, ни 11, а 10 и 11 различаются уже во втором знаке. Приёмник читает биты по одному и, как только набранное совпало со словом из словаря, записывает букву и начинает новое слово. Цепочка 100110: 1 – такого слова нет, 10 – Б; 0 – А; 1, 11 – В; 0 – А. Получилось БАВА.

Проверять правило удобно с коротких слов. Возьми самое короткое кодовое слово и посмотри, не начинается ли с него какое-нибудь длинное. Потом – следующее по длине. Нашлось хоть одно такое слово – жди путаницы, как с ББ и В на картинке.

Запомни. Равномерный код режут на куски одной длины. Неравномерный читается без разделителей, если ни одно кодовое слово не служит началом другого. Читают его по одному биту: совпало со словом из словаря – отрезай.

4.8. Разбираем задачи

Задача 4.1 Восемь светодиодов

разминка · двоичная система, перевод в обе стороны

На плате-тренажёре восемь светодиодов в ряд показывают байт: горит – 1, не горит – 0. Сейчас слева направо они горят так: горит, горит, не горит, горит, не горит, не горит, горит, горит. а) Какое число показано? б) Какие светодиоды надо зажечь, чтобы показать число 75? Ответ запиши байтом из восьми цифр.

Дано: байт 11010011_2 ; число 75.

Найти: десятичное значение байта; 75 в виде байта.

Решение

Шаг 1. Записываем байт. Горит – 1, не горит – 0: 11010011 .

Шаг 2. Подписываем веса. Справа налево: 1, 2, 4, 8, 16, 32, 64, 128. Значит, слева направо цифры стоят так: 1 под весом 128, 1 под 64, 0 под 32, 1 под 16, 0 под 8, 0 под 4, 1 под 2, 1 под 1.

Шаг 3. Складываем веса под единицами. $128 + 64 + 16 + 2 + 1 = 211$.

Шаг 4. Набираем 75 гирями. Гиря 128 больше 75 – не берём. 64 подходит: $75 - 64 = 11$. 32 и 16 больше 11 – пропускаем. 8 подходит: $11 - 8 = 3$. 4 больше 3 – пропускаем. 2 подходит: $3 - 2 = 1$. 1 подходит: $1 - 1 = 0$. Взяты гири 64, 8, 2, 1.

Шаг 5. Пишем байт. Места слева направо: 128, 64, 32, 16, 8, 4, 2, 1. Единицы – под 64, 8, 2 и 1: 01001011 . Зажечь нужно второй, пятый, седьмой и восьмой светодиоды слева.

Шаг 6. Проверяем делением. $75 : 2 = 37$ (ост. 1), $37 : 2 = 18$ (ост. 1), $18 : 2 = 9$ (ост. 0), $9 : 2 = 4$ (ост. 1), $4 : 2 = 2$ (ост. 0), $2 : 2 = 1$ (ост. 0), $1 : 2 = 0$ (ост. 1). Снизу вверх: 1001011 – семь цифр, слева дописываем ноль до байта. Совпало с шагом 5.

Первая цифра байта 0 – значит, число меньше 128. У 75 так и есть. А байт из первого вопроса начинается с двух единиц: $128 + 64 = 192$, число должно быть не меньше 192 – и 211 не меньше.

⚠ Типичная ошибка. Подписать веса слева направо – считать первую цифру за 1, вторую за 2 и так далее. Тогда 11010011 превращается в $1 + 2 + 8 + 64 + 128 = 203$. Во втором вопросе теряют ведущий ноль и пишут 1001011 : число то же, но это семь цифр, а просили байт.

Ответ: а) 211; б) 01001011 .

Своя задача по программе сферы, модуль 4

Реши сам

4.1. Светодиоды показывают байт `01100101`. Какое это число?

4.2. Лёша передал Юле записку с байтом `10011010`, а Юля прочитала его справа налево. Какое число записал Лёша, какое прочитала Юля и на сколько они отличаются?

Задача 4.2 Код цвета

шестнадцатеричная система, код цвета

В программе для рисования цвет задан кодом `#1BAAE3`. а) Сколько в нём красного, зелёного и синего? Ответ – три десятичных числа от 0 до 255. б) Художнику нужен оранжевый: красного 255, зелёного 140, синего 0. Какой код цвета ему ввести?

Дано: код `#1BAAE3`; $R = 255$, $G = 140$, $B = 0$.

Найти: R , G , B для первого цвета; код второго цвета.

Решение

Шаг 1. Режем код на байты. После решётки шесть цифр – три байта по две цифры: `1B` – красный, `AA` – зелёный, `E3` – синий.

Шаг 2. Красный. $1B_{16}$: старшая цифра 1, младшая B = 11. $1 \cdot 16 + 11 = 27$.

Шаг 3. Зелёный. AA_{16} : A = 10. $10 \cdot 16 + 10 = 170$.

Шаг 4. Синий. $E3_{16}$: E = 14. $14 \cdot 16 + 3 = 224 + 3 = 227$.

Шаг 5. Обратно: красный 255. Делим на 16 с остатком: $255 = 15 \cdot 16 + 15$. Частное 15 – это F, остаток 15 – тоже F. Получаем `FF`.

Шаг 6. Зелёный 140. $140 = 8 \cdot 16 + 12$ (потому что $8 \cdot 16 = 128$, и $140 - 128 = 12$). Частное 8, остаток 12 – это C. Получаем `8C`. Проверим через двоичную: $140 = 128 + 8 + 4 = 10001100_2$, четвёрки $1000 = 8$ и $1100 = C$.

Шаг 7. Синий 0. Это тоже байт, и он записывается двумя цифрами: `00`.

Шаг 8. Собираем код. `#FF8C00`.

В первом цвете синего больше всего (227 из 255), зелёного тоже много, а красного почти нет. Такой цвет выйдет голубым, как летнее небо над Обским морем. Можешь ввести оба кода в любой графический редактор и посмотреть.

⚠ Типичная ошибка. Забыть, что буквы – это числа: считать B второй буквой и ставить 2 или путать $B = 11$ и $C = 12$. Вторая частая ошибка – записать ноль синего одной цифрой: `#FF8C0`. Это пять цифр, и программа такой код не поймёт. И ещё: при делении 140 на 16 калькулятор выдаёт 8,75, и непонятно, что с этим делать. Нужны частное и остаток: $140 = 8 \cdot 16 + 12$, а дробная часть 0,75 – это 12:16.

Ответ: а) красный 27, зелёный 170, синий 227; б) `#FF8C00`.

Реши сам

4.3. Байт записан в шестнадцатеричной системе: 7F. Запиши его в двоичной системе (8 цифр) и в десятичной.

4.4.* Самодельный приёмник – цепочка стрелочных индикаторов. Каждая стрелка показывает число от 0 до 15, то есть очередные четыре бита сигнала. Две стрелки подряд – один байт, а байт – номер латинской заглавной буквы в таблице ASCII (A = 65, B = 66 и дальше подряд до Z = 90). Приёмник включили посреди передачи, поэтому одно показание лишнее – либо первое, либо последнее. Стрелки показали: 3, 4, 13, 4, 9, 5, 3, 4, 8, 4, 1. Какое слово передали?

Задача 4.3 Команды для робота

сколько бит нужно, 2 в степени n, объём сообщения

Для учебного робота придумали 20 разных команд. Каждая передаётся равномерным двоичным кодом – все кодовые слова одной длины. а) Сколько бит нужно на одну команду? б) Сколько новых команд можно добавить, не удлиняя кодовые слова? в) Сколько бит займёт программа из 45 команд?

Дано: $M = 20$ команд; код равномерный; в программе $K = 45$ команд.

Найти: i – ?; сколько свободных кодов; I – ?

Решение

Шаг 1. Перебираем степени двойки. $2^4 = 16$ – меньше 20, четырёх бит не хватит. $2^5 = 32$ – не меньше 20, хватает. Значит, $i = 5$ бит.

Шаг 2. Свободные коды. Пятью битами можно записать 32 разных слова, заняты 20: $32 - 20 = 12$ свободных. Столько команд можно добавить, пока код не придётся удлинять.

Шаг 3. Объём программы. $I = K \cdot i = 45 \cdot 5 = 225$ бит.

В байтах это $225 : 8 \approx 28$ байт. Фотография в телефоне весит миллионы байт, так что программа для робота – крошка, её можно передать даже медленным каналом.

 **Типичная ошибка.** Взять 4 бита, потому что 20 «ближе» к 16, чем к 32. Округлять здесь можно только вверх: четыре бита дают 16 кодов, и четырём командам кода не достанется. А в вопросе б) часто отвечают «32» – но 32 – это все коды, свободных среди них $32 - 20 = 12$.

Ответ: а) 5 бит; б) 12 команд; в) 225 бит.

Реши сам

4.5. В одной старой таблице символов на каждый знак отводили 7 бит. Сколько разных знаков в ней помещалось?

4.6. Датчик в теплице меряет температуру целыми градусами от -20 до $+40$ °C и передаёт каждое показание равномерным двоичным кодом. а) Сколько бит нужно на одно показание? б) Сколько байт наберётся за сутки, если датчик меряет температуру раз в 10 минут?

Задача 4.4 Сообщение по ИК-лучу

скорость передачи, время передачи, длительность бита

ИК-передатчик отправляет сообщение из 75 символов, на каждый символ приходится 8 бит. Скорость передачи – 400 бит/с. а) За сколько секунд уйдёт сообщение? б) Сколько миллисекунд длится один бит?

Дано: $K = 75$ символов; $i = 8$ бит; $v = 400$ бит/с.

Найти: t – ?; τ – ? (в мс)

Решение

Шаг 1. Объём в битах. Скорость дана в битах в секунду, значит, и сообщение считаем в битах: $I = 75 \cdot 8 = 600$ бит.

Шаг 2. Время. $t = I : v = 600 : 400 = 1,5$ с.

Шаг 3. Длительность бита. $\tau = 1 : v = 1 : 400 = 0,0025$ с.

Шаг 4. Секунды в миллисекунды. В секунде 1000 мс: $0,0025 \cdot 1000 = 2,5$ мс.

Проверим с другой стороны: 600 бит по 2,5 мс – это $600 \cdot 2,5 = 1500$ мс, те же 1,5 с. Полторы секунды на строчку текста – медленно по меркам Wi-Fi, но ИК-луч и не возит большие файлы: его дело – короткие команды и сообщения.

⚠ Типичная ошибка. Поделить 75 на 400 и забыть, что символ – это 8 бит: выйдет 0,19 с, в 8 раз меньше верного. Во втором вопросе легко записать 0,0025 – но это секунды, а ответ просили в миллисекундах.

Ответ: а) 1,5 с; б) 2,5 мс.

Своя задача по программе сферы, модуль 4

Реши сам

4.7. Один бит в канале длится 20 мс. Какова скорость передачи в бит/с? Сколько бит пройдёт по каналу за 3 с?

4.8. Значок для экрана – картинка 16×16 точек, в его палитре 16 цветов. Значок передают по каналу со скоростью 256 бит/с. Сколько секунд займёт передача?

Задача 4.5 Где испорчен байт

бит чётности, контрольная сумма, поиск ошибки

а) Приёмник получил четыре байта. К каждому передатчик дописал бит чётности, так что в каждой девятке бит единиц должно быть чётное число: 010010110, 110100100, 001100001, 101010100. Какой байт по счёту пришёл с ошибкой? б) В другом сообщении передали четыре числа и контрольную сумму – остаток от деления их суммы на 256. Приёмник получил числа 45, 120, 17, 200 и контрольную сумму 116. Есть ли в сообщении ошибка? Ответ: номер байта; «да» или «нет».

Дано: четыре девятки бит с битом чётности; числа 45, 120, 17, 200; контрольная сумма 116, правило – остаток от деления суммы на 256.

Найти: номер испорченного байта; есть ли ошибка во втором сообщении.

Решение

Шаг 1. Считаем единицы в каждой девятке – вместе с битом чётности.

- 010010110 – 4 единицы, чётно;
- 110100100 – 4 единицы, чётно;
- 001100001 – 3 единицы, **нечётно**;
- 101010100 – 4 единицы, чётно.

Испорчен третий байт. Какой бит в нём перевернулся, мы не знаем, – приёмник попросит прислать этот байт ещё раз.

Шаг 2. Складываем полученные числа. $45 + 120 + 17 + 200 = 382$.

Шаг 3. Остаток от деления на 256. 382 больше 256, но меньше $2 \cdot 256 = 512$, поэтому вычитаем 256 один раз: $382 - 256 = 126$.

Шаг 4. Сравниваем. Приёмник насчитал 126, а в сообщении пришло 116. Не совпало – значит, ошибка есть. Какое из пяти чисел испорчено, контрольная сумма не говорит: разница в 10 могла прийти откуда угодно, даже из самой суммы. (Если передавали 7 вместо 17, то у передатчика было $45 + 120 + 7 + 200 = 372$ и $372 - 256 = 116$ – как раз присланная сумма.)

 **Типичная ошибка.** Считать единицы только в первых восьми битах и забыть про бит чётности. Тогда у третьего байта 00110000 две единицы – чётно, и ошибка ускользает. Во втором вопросе забывают взять остаток и сравнивают 382 со 116. Вывод «не равно» случайно верный, но при таком подсчёте и правильное сообщение (сумма 372) не совпало бы со 116.

Ответ: а) третий; б) да, ошибка есть: контрольная сумма получилась 126, а пришла 116.

Своя задача по программе сферы, модуль 4

Реши сам

4.9. а) Допиши бит чётности к семибитным словам 1110110 и 0100000, чтобы единиц стало чётное число. б) Передают числа 200, 100 и 50. Посчитай контрольную сумму – остаток от деления их суммы на 256.

4.10.* Четыре шестнадцатеричные цифры передали таблицей 4 × 4 бита: каждая строка – одна цифра, в конце строки – бит чётности этой строки, в последней строке – биты чётности столбцов. Известно, что по дороге испортился один бит. Пришло:

1	1	0	0		0
0	0	0	0		0
1	0	0	1		1
1	1	1	0		1
-----	+	---			
1	1	1	1		0

Найди испорченный бит, исправь его и запиши четыре переданные цифры подряд.

Задача 4.6 Телеграфист со своим почерком

азбука Морзе, длительности, время передачи

Опытный телеграфист передаёт азбуку Морзе в своём темпе. Все длительности у него всегда одинаковые, но не такие, как в справочнике. Приёмник записал его сигнал (рисунок ниже): каждая клетка – 0,1 с, закрашенная клетка – звучит тон. а) Какое слово передано? б) Сколько секунд длилась передача?



Рисунок 4.10 – Запись сигнала телеграфиста. Одна клетка – 0,1 с, закрашено – звучит тон

Дано: запись сигнала, 30 клеток по 0,1 с.

Найти: слово; t – ?

Решение

Шаг 1. Какие бывают тоны. Считаем закрашенные клетки подряд: встречаются тоны длиной 1 клетка и 2 клетки. Короткий – точка, длинный – тире. У этого телеграфиста тире вдвое длиннее точки, а не втрое, как в справочнике. Для расшифровки это неважно: тире всё равно длиннее.

Шаг 2. Какие бывают паузы. Пустые промежутки между тонами – 1 клетка или 3 клетки. Короткая пауза разделяет знаки внутри буквы, длинная – буквы.

Шаг 3. Делим на буквы. Длинных пауз две, значит, букв три. Первая группа: короткий, длинный, короткий, короткий – **.-..**. Вторая: короткий, короткий, длинный – **..-**. Третья: длинный, длинный, длинный, короткий – **----.**

Шаг 4. Ищем буквы в справочнике. **.-..** – Л, **..-** – У, **----.** – Ч. Слово – ЛУЧ.

Шаг 5. Время. Вся запись от первого тона до последнего – 30 клеток. Пересчитаем по частям: Л – $1 + 1 + 2 + 1 + 1 + 1 + 1 = 8$ клеток, пауза 3, У – $1 + 1 + 1 + 1 + 2 = 6$, пауза 3, Ч – $2 + 1 + 2 + 1 + 2 + 1 + 1 = 10$. Всего $8 + 3 + 6 + 3 + 10 = 30$ клеток, это $30 \cdot 0,1 = 3$ с. Выходит буква в секунду – спокойный темп, на слух такое разберёт и новичок.

⚠ Типичная ошибка. Искать по справочнику тоны «втрое длиннее точки» и не найти ни одного – а потом решить, что в записи одни точки. Длительности в Морзе – договорённость двух людей: тире длиннее точки, пауза между буквами длиннее паузы внутри буквы, остальное зависит от почерка. Вторая ошибка – считать время только по тонам и забыть паузы: 16 клеток, 1,6 с.

Ответ: а) ЛУЧ; б) 3 с.

По мотивам задачи второго отборочного этапа НТО 2025/26 (трек 8–11 классов), профиль «Технологии беспроводной связи», «Телеграф Премиум»

Реши сам

4.11. Сигнал бедствия SOS передают как три буквы: С (...), О (---), С (...) – с паузами между буквами по стандартным правилам из раздела 4.6. Сколько единиц времени длится один сигнал? Сколько это секунд, если точка длится 0,1 с?

4.12. Расшифруй сообщение. Между буквами стоит пробел, между словами – косая черта: .. -.. .. / -.. --- -- --- .----

Задача 4.7* Свой код для машинки

неравномерный код, декодирование без разделителей, свой алфавит кода

Юля и Миша придумали для радиоуправляемой машинки неравномерный код: ПРЯМО – 0, НАЗАД – 10, НАЛЕВО – 110, НАПРАВО – 111. а) Пришла цепочка 0010011000111. Какие команды в ней? Запиши их подряд буквами П (прямо), З (назад), Л (налево), Р (направо). б) Сколько бит заняли бы те же команды в равномерном коде и сколько бит сэкономил неравномерный? в) Миша предлагает укоротить НАПРАВО до 11. Можно ли так сделать?

Дано: П – 0, З – 10, Л – 110, Р – 111; цепочка 0010011000111.

Найти: команды; объём в равномерном коде и экономию; можно ли Р – 11.

Решение

Шаг 1. Проверяем правило. Ни одно слово не начинается другое: 0 – единственное слово на ноль, 10 отличается от 110 и 111 уже вторым знаком, а 110 и 111 – третьим. Удобно нарисовать код деревом: от каждой развилки две дороги – 0 и 1, а команды сидят только на концах веток. Если ни одна команда не стоит на середине пути к другой, код читается без разделителей.

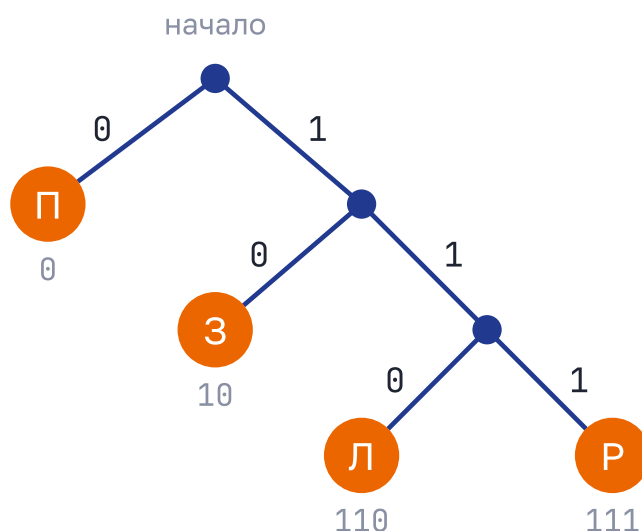


Рисунок 4.11 – Дерево кода: каждая развилка – очередной бит, команды стоят только на концах веток

Шаг 2. Читаем цепочку по одному биту. Набираем биты, пока не совпадёт слово из словаря:

- 0 – П;
- 0 – П;
- 1 – такого слова нет, 10 – 3;
- 0 – П;
- 1, 11 – слов нет, 110 – Л;
- 0 – П;
- 0 – П;
- 1, 11, 111 – Р.

Биты закончились вместе с последним словом, лишних не осталось. Команды: ППЗПЛППР.

Шаг 3. Равномерный код. Команд 4, это 2^2 , – хватит двух бит на команду. В цепочке 8 команд: $8 \cdot 2 = 16$ бит.

Шаг 4. Экономия. В нашей цепочке 13 бит: $16 - 13 = 3$ бита. Выигрыш получился потому, что ПРЯМО – самая частая команда (5 раз из 8), и ей достался код из одного бита. Если бы машинка чаще поворачивала, неравномерный код мог бы и проиграть.

Шаг 5. Можно ли НАПРАВО – 11 ? Тогда 11 станет началом слова 110. Приёмник получит 110 и не поймёт: это НАЛЕВО или НАПРАВО, а потом ПРЯМО? На дереве команда Р оказалась бы на середине пути к Л. Так делать нельзя.

⚠ Типичная ошибка. Резать цепочку по два бита, как равномерный код: 00, 10, 01, 10, 00, 11 – и ещё один бит лишний. Выходит ерунда, а таких слов, как 00 или 01, в словаре вообще нет. Неравномерный код читают только по одному биту, каждый раз сверяясь со словарём.

Ответ: а) ППЗПЛППР; б) 16 бит, экономия 3 бита; в) нельзя: 11 – начало слова 110.

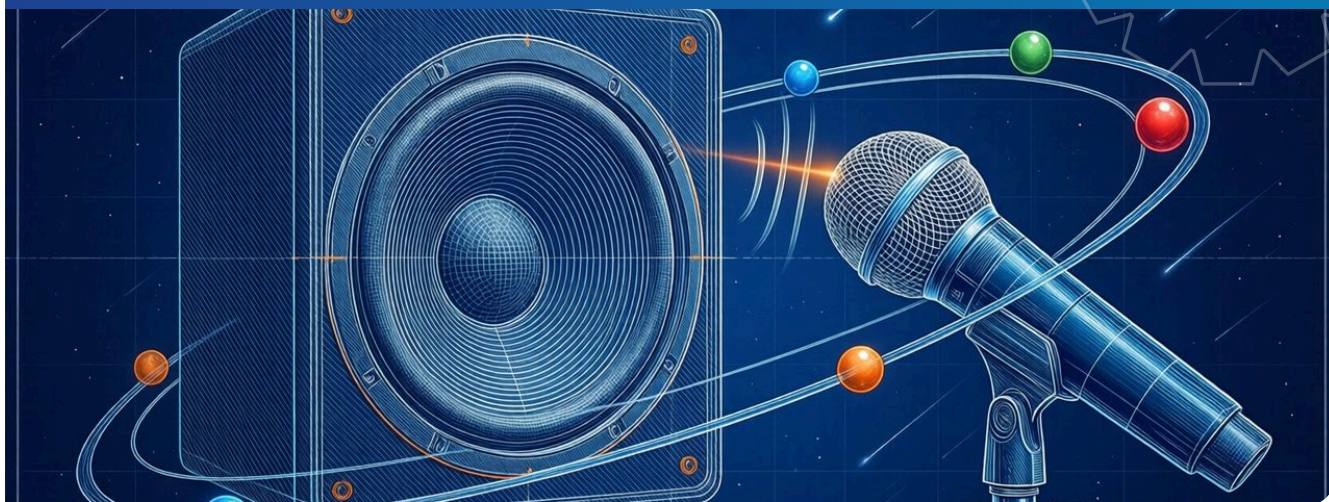
Своя задача по программе сферы, модуль 4

Реши сам

4.13. Код: А – 1, Б – 01, В – 001, Г – 000. Расшифруй цепочку 1010011000.

4.14.* Коля придумал код: А – 0, Б – 01, В – 10. Ему пришла цепочка 010. а) Какие два разных слова могли ему передать? б) Коля хочет исправить код: к одному из трёх кодовых слов дописать в конце одну цифру так, чтобы ни одно слово не было началом другого. Какое слово и как надо изменить?

Модуляция и каналы связи



Нули и единицы из прошлой главы пора отправить в путь – звуком от динамика к микрофону или невидимым инфракрасным лучом, как у пульта от телевизора. Здесь биты «сажают» на несущую, а потом снимают обратно по графику. На отборе такая задача приходит с графиком сигнала, таблицей или парой чисел. Ответ – строка из нулей и единиц, номера подходящих вариантов или время в секундах.

5.1. Зачем нужна модуляция

НА ПАЛЬЦАХ



Рисунок 5.1 – Несущая везёт сообщение, как грузовик посылку

Посылка сама до другого города не доедет, её везёт грузовик. Сообщение тоже не летит само: его везёт несущая.

Команда для КиберМишки «вперёд» хранится в памяти как два бита, например 00. Но по воздуху не может полететь «ноль» или «единица». Полететь может звук – колебания воздуха – или свет. Значит, биты нужно превратить в звук или в свет так, чтобы на другом конце их можно было узнать.

Делают это так. Передатчик издаёт ровный тон одной частоты – **несущую** (мы познакомились с ней в главе 3). Сама по себе несущая – просто писк, сообщения в нём нет. Потом передатчик начинает менять у несущей какой-нибудь параметр в такт сообщению: для единицы – одно, для нуля – другое. Такое изменение несущей по сообщению называют **модуляцией**. Приёмник делает обратную работу: смотрит, как менялась несущая, и восстанавливает биты. Это **демодуляция**.

Зачем эта морока с несущей, почему не передать биты как есть?

- **Среда пропускает не всё.** Динамик и микрофон хорошо работают только в своём диапазоне частот. Слишком низкий гул маленький динамик просто не издаст. Несущую выбирают там, где канал работает лучше всего.
- **Много передатчиков в одном месте.** В Новосибирске десятки радиостанций, и все вещают одновременно. Каждая – на своей несущей частоте, поэтому ты крутишь ручку приёмника и выбираешь одну из них. Без несущих они слились бы в кашу.
- **Шум.** Приёмник, настроенный на одну частоту, всё остальное отбрасывает. Инфракрасный свет пульта от телевизора у многих моделей мигает около 38 000 раз в секунду, и приёмник в телевизоре ловит только такое мигание. Ровный свет солнца или лампы он просто не замечает.

Вспомни из главы 2: у колебания три параметра – **амплитуда** (размах, громкость), **частота** (сколько колебаний в секунду, высота тона) и **фаза** (сдвиг колебания во времени). Какой параметр меняем, так модуляция и называется: **амплитудная** (АМ), **частотная** (ЧМ) или фазовая. На радиоприёмнике ты видел кнопки АМ и FM – это те же самые слова по-английски: amplitude modulation и frequency modulation. Фазовую в этой книге мы только называем, задач на неё не будет.

Ещё одна договорённость нужна до начала передачи. Сообщение режут по времени на одинаковые кусочки, в каждом кусочке – один бит. Время одного кусочка – **длительность бита** τ («тау»), ты встречал её в главе 4. Передатчик и приёмник должны знать её заранее, иначе приёмник не поймёт, где кончается один бит и начинается другой. Похоже на песню: чтобы хлопнуть в такт, надо знать темп.

Запомни. Модуляция – сообщение «сажают» на несущую, меняя её амплитуду (АМ) или частоту (ЧМ). Демодуляция – приёмник достаёт сообщение обратно. Сообщение делят на кусочки одинаковой длительности τ , по одному биту в каждом.

5.2. Амплитудная модуляция: громче – тише

НА ПАЛЬЦАХ

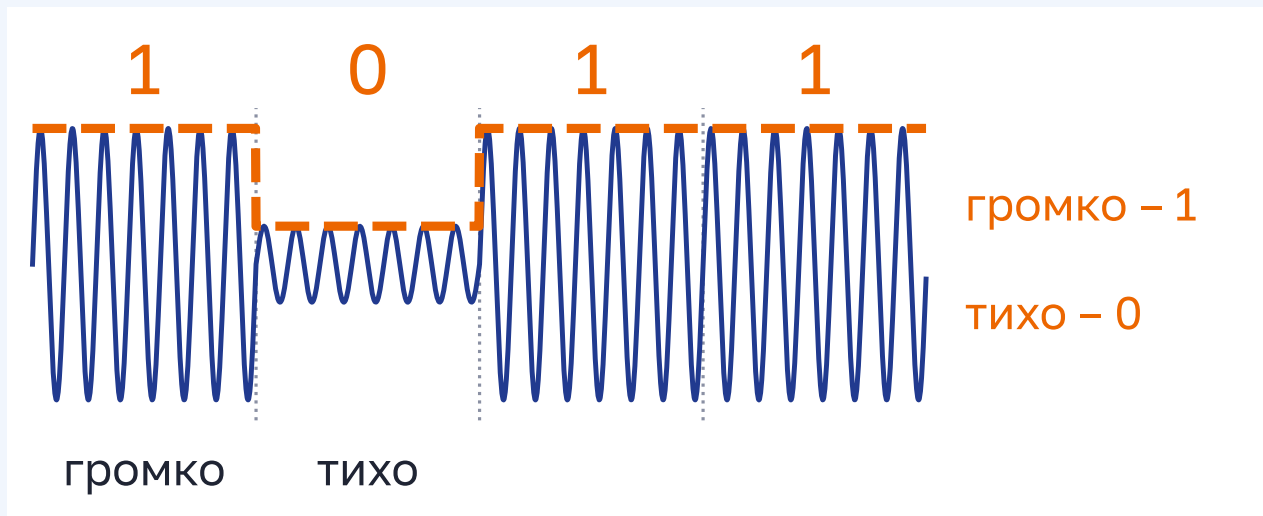


Рисунок 5.2 – Один и тот же тон то громче, то тише. Громкий кусочек – единица, тихий – ноль

Частота всё время одна и та же, меняется только размах. Оранжевая линия по верхушкам – огибающая.

При **амплитудной модуляции** передатчик меняет громкость несущей: в кусочке с единицей тон звучит громко, в кусочке с нулём – тихо. Часто ноль передают вообще тишиной: тон есть – 1, тона нет – 0. Так в походе передают азбуку Морзе фонариком, так же в старину работал телеграф.

Приёмник читает такой сигнал так.

- **Разрежь запись на кусочки** длиной τ , начиная с момента, когда началось сообщение. Если бит длится 10 мс, границы будут на 10, 20, 30 мс и так далее.
- **В каждом кусочке посмотри на размах** колебаний – на огибающую. Большой размах – 1, маленький – 0. Если сомневаешься, сравни с порогом, как в главе 2: порог ставят посередине между «громко» и «тихо».
- **Выпиши биты** по порядку.

Коварное место одно – одинаковые биты подряд. Три единицы подряд на графике выглядят как одна длинная громкая пачка. Приёмник не видит, где кончилась первая единица и началась вторая, он просто делит длину пачки на длительность бита:

$$\text{число бит в пачке} = \frac{\text{длительность пачки}}{\tau}.$$

Пачка тона длиной 0,3 с при $\tau = 0,1$ с – это $0,3 : 0,1 = 3$ единицы. Тишина длиной 0,2 с – два нуля.

Сколько колебаний несущей помещается в один бит? Частота показывает, сколько колебаний проходит за секунду, значит, за время τ их пройдёт

$$n = f \cdot \tau,$$

где f – частота несущей в герцах, τ – длительность бита в секундах. Несущая 1000 Гц, бит 10 мс: сначала 10 мс переводим в секунды, это 0,01 с, потом $1000 \cdot 0,01 = 10$ колебаний. На

графике каждая громкая или тихая пачка у такого бита будет состоять из десяти колебаний.

У АМ есть слабое место: громкость меняет не один передатчик. КиберМишка отъехал по-дальше – и «громко» стало тише. Между динамиком и микрофоном поставили картонную преграду – ещё тише. В классе хлопнула дверь – громко там, где передатчик молчал. Порог, который хорошо работал вблизи, вдали начинает ошибаться. На финале сферы на поле ставят преграды, и порог, подобранный без них, может подвести.

Запомни. АМ: громко – 1, тихо (или тишина) – 0. Запись режут на кусочки по τ и в каждом смотрят размах. Длинная пачка – несколько одинаковых бит подряд: длительность пачки делим на τ . Колебаний несущей в одном бите: $n = f \cdot \tau$.

Совет. Когда читаешь график, карандашом проведи вертикальные линии через каждые τ и подпиши под каждым кусочком 0 или 1. Глазами по длинной пачке очень легко «потерять» одну единицу, а по разлинованной записи – нет.

5.3. Частотная модуляция: выше – ниже тон

НА ПАЛЬЦАХ

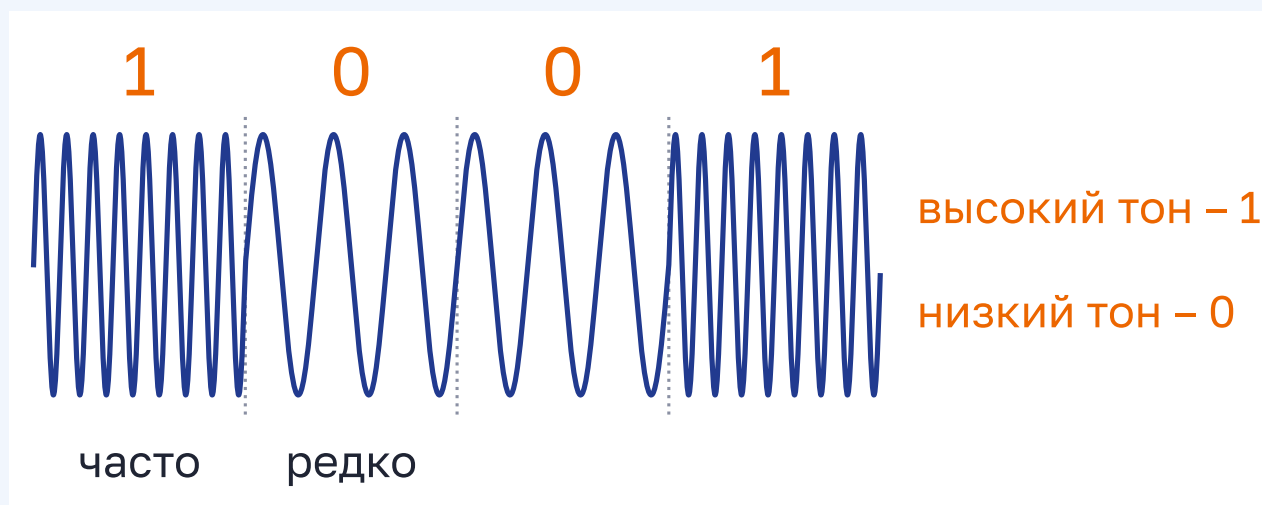


Рисунок 5.3 – Громкость всё время одинаковая, меняется высота тона

Как свисток с двумя нотами: тонкий писк – единица, басовитый гудок – ноль.

При **частотной модуляции** громкость остаётся прежней, меняется высота тона. Для единицы передатчик свистит на одной частоте, для нуля – на другой. Договариваются заранее: например, 1 – 2000 Гц, 0 – 1000 Гц. На графике кусочек с высоким тоном выглядит густым – колебания идут часто, с низким – редким.

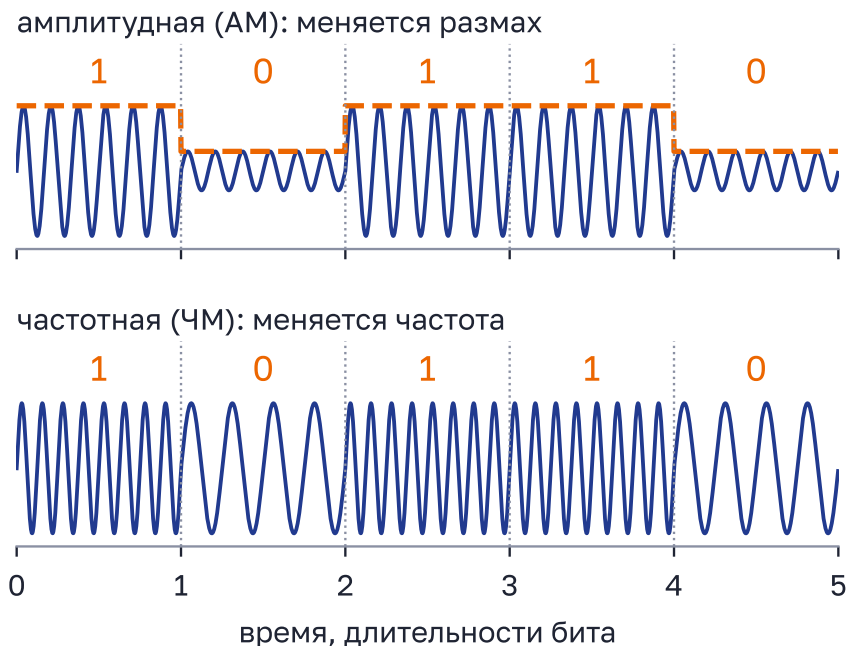


Рисунок 5.4 – Одни и те же биты 1 0 1 1 0, переданные двумя способами

Приёмник узнаёт частоту по периоду, как велокомпьютер из главы 2. Можно сосчитать колебания в кусочке и поделить на длительность бита:

$$f = \frac{n}{T}.$$

В кусочке 10 мс насчитали 6 колебаний. 10 мс = 0,01 с, частота $6 : 0,01 = 600$ Гц. А можно измерить период одного колебания T и найти частоту как $f = 1 : T$: период 0,5 мс = 0,0005 с даёт $1 : 0,0005 = 2000$ Гц.

Колебания считай по вершушкам: одна вершушка – одно колебание. Если считать и вершушки, и впадины, получится вдвое больше.

Чем ЧМ лучше АМ? По дороге звук теряет громкость, высота тона остаётся прежней. Кибер-Мишка отъехал на другой конец класса – свист стал тише, но остался таким же тонким. Шум тоже в основном меняет громкость, поэтому ЧМ сбивается реже. Платить за это приходится сложностью: приёмнику мало отличить «громко» от «тихо», он должен различать частоты.

динамик и микрофон работают хорошо

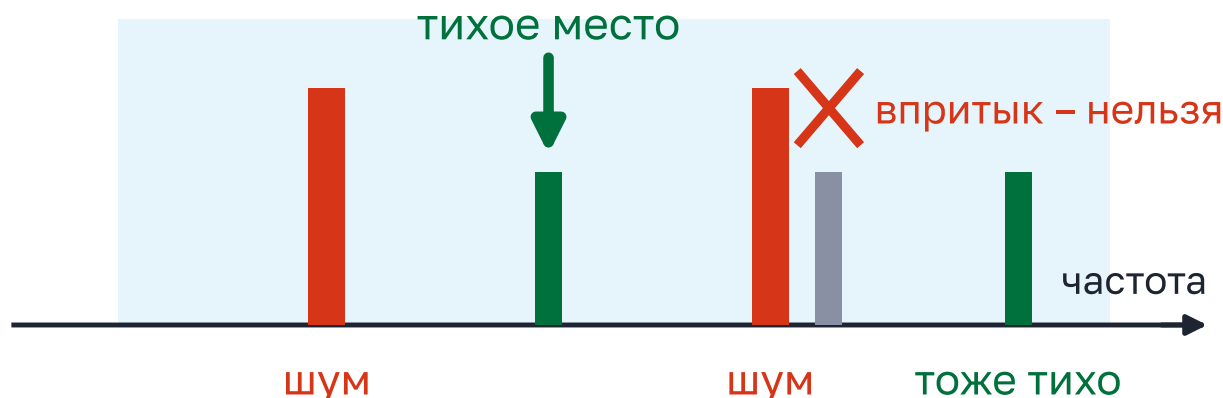


Рисунок 5.5 – Шум стоит на своих частотах. Свою частоту ставим в тихое место между ними

Как место для разговора в шумной столовой: садись подальше от окна раздачи, где гремят подносы.

Как выбрать частоты для ЧМ. В задачах проверяют четыре правила.

- Обе частоты должны попадать в **диапазон**, где хорошо работают и динамик, и микрофон.
- Они должны быть **подальше от шума**. В главе 3 мы разбирали спектр: если в кабинете гудит вентилятор на 1200 Гц, частоту 1250 Гц приёмник спутает с гулом.
- Частоты нуля и единицы должны быть **подальше друг от друга**, иначе приёмник их перепутает.
- Удобно, когда в одном бите помещается **целое число колебаний** ($n = f \cdot \tau$ – целое) и их не слишком мало. По двум-трём колебаниям частоту уверенно не узнаешь.

Запомни. ЧМ: высокий тон – одно значение бита, низкий – другое, какое именно – договариваются заранее. Частота по графику: $f = n : \tau$ или $f = 1 : T$. ЧМ меньше боится расстояния и шума, чем АМ. Частоты выбирают внутри рабочего диапазона, вдали от шума и друг от друга.

Для любопытных. Когда модулируют цифровым сообщением – нулями и единицами, – инженеры часто говорят не «модуляция», а **манипуляция**: амплитудная манипуляция, частотная манипуляция. Слово пугает, но ничего нового в нём нет: это те же АМ и ЧМ, только параметр прыгает между двумя значениями, а не меняется плавно, как у голоса по радио.

5.4. Канал связи: звук и инфракрасный луч

НА ПАЛЬЦАХ

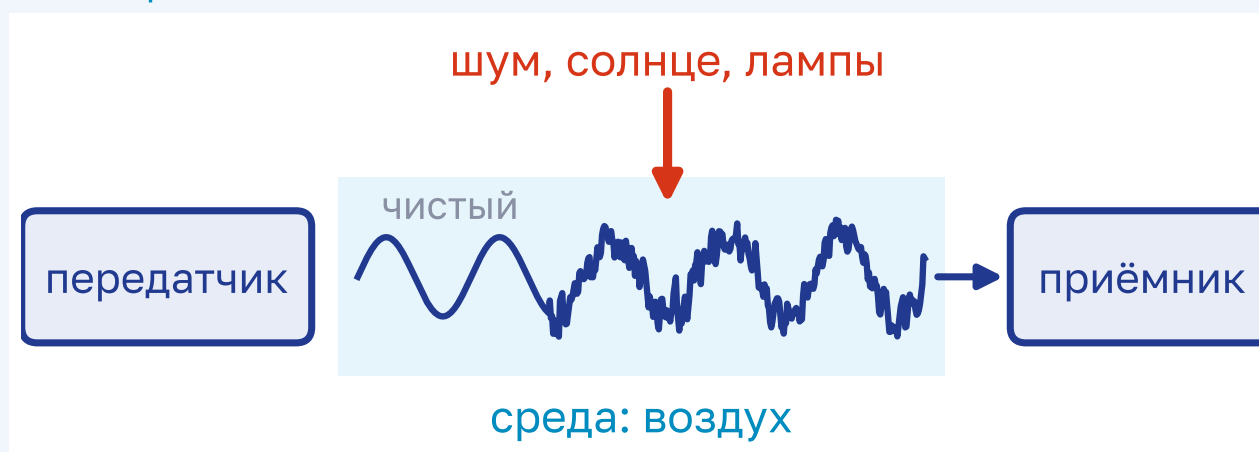


Рисунок 5.6 – Сигнал уходит от передатчика чистым, а по дороге к нему прилипают помехи

Среда – это дорога. Сигнал по ней не только едет, но и пачкается.

В главе 1 мы раскладывали канал связи на пять частей: источник, кодирующее устройство, среда, декодирующее устройство, получатель. В этой главе удобнее говорить короче: **передатчик** (превращает биты в модулированный сигнал), **среда** (то, через что сигнал идёт) и **приёмник** (достаёт из сигнала биты). В сфере «Кибертех» два главных канала.

Акустический канал. Передатчик – динамик или излучатель звука, среда – воздух, приёмник – микрофон. В комплекте «ТЮК-Акустика» есть излучатель и микрофоны, а КиберМишка умеет принимать звуковой сигнал. Звук обходит небольшие преграды: голос друга слышно из-за угла. Зато звук медленный, и ему мешает любой шум в комнате.

ИК-канал. Передатчик – инфракрасный светодиод, приёмник – фотоприёмник, который чувствует инфракрасный свет. Так работает пульт от телевизора. КиберМишка умеет принимать и ИК-сигнал, а для турниров есть комплект «Инфракрасный сигнал». Свет летит почти мгновенно, но через непрозрачную преграду не проходит. Нужна прямая видимость или отражение: пульт иногда срабатывает, даже если направить его в стену.

Таблица 5.1

	Звук	ИК-луч
Передатчик	динамик	ИК-светодиод
Приёмник	микрофон	фотоприёмник
Скорость в воздухе	около 340 м/с	около 300 000 км/с
Преграда	обходит, но слабеет	нужна прямая видимость
Что мешает	шум, эхо	солнце, лампы



Рисунок 5.7 – В грозу вспышку видно сразу, а гром приходит позже. Звук едет медленно, свет – почти мгновенно

Досчитай от вспышки до грома: каждые три секунды – это примерно километр до грозы.

Задержка. Сигналу нужно время, чтобы долететь. Время равно расстоянию, делённому на скорость: $t = s : v$. Через спортзал длиной 17 м звук летит $17 : 340 = 0,05$ с, то есть 50 мс. Если бит длится 10 мс, за это время передатчик успеет выпустить пять бит, пока первый ещё в пути. ИК-луч те же 17 м пролетит примерно за 0,00000006 с – для наших задач это ноль.

Помехи. Всё лишнее, что попало в сигнал по дороге, – **помеха**, или шум (о шуме мы говорили в главе 2). Для звука это голоса, хлопки, гул вентилятора и **эхо**: звук отражается от стен и приходит второй раз, с опозданием, прямо поверх следующих бит. Для ИК-луча помеха – солнце и лампы, они ведь тоже светят в инфракрасном. С помехами борются по-разному. Несущую ставят туда, где шума меньше. Бит делают подлиннее. А к сообщению добавляют проверочные биты – о них в следующем разделе.

Запомни. Канал: передатчик → среда → приёмник. Звук: 340 м/с, огибает преграды, боится шума и эха. ИК: почти мгновенно, нужна прямая видимость, мешают солнце и лампы. Задержка в пути $t = s : v$.

5.5. Скорость, время и надёжность

НА ПАЛЬЦАХ



Рисунок 5.8 – Каждый бит послали трижды. Помеха испортила один из трёх – приёмник всё равно понял

Как в классе, когда плохо слышно: просишь повторить три раза и веришь тому, что услышал два раза из трёх.

Скорость передачи и время мы уже считали в главе 4. Напомним в двух строчках: скорость v – сколько бит проходит за секунду, один бит длится $\tau = 1 : v$, а сообщение из I бит идёт

$$t = \frac{I}{v} = I \cdot \tau.$$

Бит 20 мс = 0,02 с – скорость 1 : 0,02 = 50 бит/с. Скорость 200 бит/с – бит длится 1 : 200 = 0,005 с = 5 мс.

В настоящей передаче кроме самого сообщения по каналу едут служебные биты, и в I нужно сложить всё, что прозвучит или просветит.

- **Стартовая метка** – условная комбинация в начале, например 10101010. По ней приёмник понимает: сообщение пошло, резать запись на кусочки надо отсюда.
- **Паузы** между частями сообщения. Помни про забор из главы 2: между 20 частями 19 промежутков, а не 20.
- **Проверочные биты** – бит чётности из главы 4 или повтор, о котором ниже.

Пример. 40 бит данных, стартовая метка 8 бит, скорость 50 бит/с. Всего 40 + 8 = 48 бит, время 48 : 50 = 0,96 с. Для звука добавь ещё задержку в пути, если расстояние большое: сообщение закончит приходить на S : $v_{\text{звука}}$ позже, чем закончит звучать.

Надёжность. Идеальных каналов не бывает: помеха может превратить 0 в 1 или 1 в 0. На сколько канал плох, показывает **доля ошибок** – сколько бит пришло неверно из всех отправленных, в процентах:

$$\text{доля ошибок} = \frac{\text{число ошибочных бит}}{\text{число отправленных бит}} \cdot 100 \, \%.$$

Чтобы её узнать, передатчик отправляет известное сообщение, а приёмник возвращает то, что получил. Два ряда сравнивают бит за битом. Из 50 бит не совпали 2 – доля ошибок 2 : 50 · 100 = 4 %.

Чем бороться с ошибками? Средства есть, но за каждое чем-то платишь.

- **Бит чётности** (глава 4) замечает одну ошибку в группе бит, но исправить её не может – не знает, какой бит виноват. Приёмнику остаётся попросить повторить.

- **Повтор.** Каждый бит отправляют три раза: 1 → 111, 0 → 000. Приёмник смотрит на тройку и выбирает то, чего в ней больше: 101 читается как 1, 001 – как 0. Это называют голосованием большинством. Одну ошибку в тройке такой способ исправляет сам, без переспрашивания. Две ошибки в одной тройке он уже пропустит: 100 прочитается как 0.
- **Длиннее бит.** Чем больше колебаний несущей в одном бите, тем увереннее приёмник решает, громко там или тихо, высоко или низко. Короткий щелчок помехи испортит маленький кусочек бита, остальное уцелеет.

Бесплатной надёжности не бывает. Повтор делает сообщение втрое длиннее: 40 бит данных превратятся в 120, и передача при 50 бит/с займёт $120:50 = 2,4$ с вместо 0,8 с. Бит чётности на каждые 8 бит добавит всего $40:8 = 5$ бит, зато только заметит ошибку. Инженер каждый раз решает по условию задачи, что ему дороже – скорость или надёжность.

Запомни. $t = I : v$, где I – все биты вместе со стартовой меткой, паузами и проверочными битами. Доля ошибок = ошибочные биты : отправленные · 100 %. Повтор трижды с голосованием исправляет одну ошибку в тройке, но удлинняет сообщение втрое.

Совет. Прежде чем считать время, выпиши столбиком, из чего состоит передача: «старт – 8, данные – $20 \cdot 9$, паузы – $19 \cdot 2$ ». Сложи и только потом дели на скорость. Так ничего не потеряется, и проверить себя легко.

5.6. Два канала сразу

НА ПАЛЬЦАХ

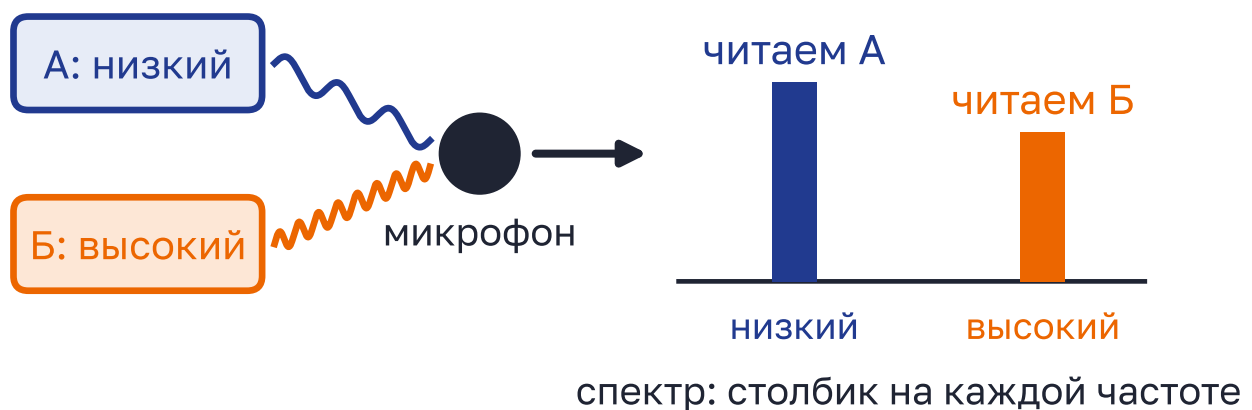


Рисунок 5.9 – Два передатчика звучат одновременно. Микрофон слышит смесь, а спектр раскладывает её по частотам

Как в хоре: бас и тонкий голос поют вместе, а ты всё равно слышишь каждого по отдельности.

В задачах финала сферы сообщение передают сразу двумя каналами – звуком и ИК-лучом. Пользуются этим двумя способами. Можно послать одно и то же по обоим каналам: подвёл один – выручит второй. А можно поделить сообщение: часть бит отправить звуком, часть лучом. Тогда всё придёт быстрее, но ждать придётся тот канал, который закончит последним.

Бывает и так, что в одной комнате одновременно звучат два передатчика, каждый на своей несущей. Микрофон слышит их смесь. Приёмник строит спектр (глава 3) и смотрит столбик на частоте каждого передатчика отдельно. Столбик на 1000 Гц растёт и падает в такт битам одного передатчика, столбик на 3000 Гц – другого. Выходит два независимых канала в одном воздухе.

Громкость у передатчиков почти никогда не одинаковая: один стоит ближе к микрофону, другой дальше. Поэтому порог «громко – тихо» подбирают для каждого столбика свой.

Запомни. Два канала сразу дают на выбор надёжность (одно и то же дважды) или скорость (сообщение поделили). Два передатчика на разных частотах разделяет спектр, и у каждого свой порог. Время поделённой передачи – по тому каналу, который закончит позже.

5.7. Разбираем задачи

Задача 5.1 Громко и тихо

разминка · амплитудная модуляция, чтение графика, длительность бита

Передатчик «ТЮК-Акустики» отправил КиберМишке 8 бит амплитудной модуляцией: громкий тон – 1, тихий – 0. Несущая – 1000 Гц, каждый бит длится 10 мс, сообщение началось в момент 0. На рисунке – запись с микрофона. Расшифруй сообщение. В ответ введи 8 цифр подряд.

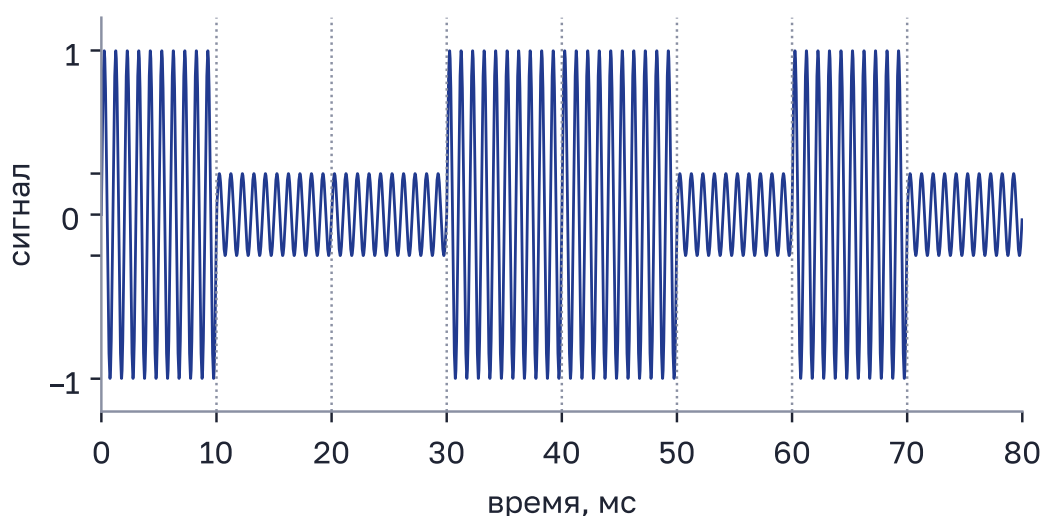


Рисунок 5.10 – Запись с микрофона. Пунктир – границы бит через 10 мс

Дано: АМ, громко – 1, тихо – 0; $f = 1000$ Гц; $\tau = 10$ мс; 8 бит.

Найти: биты – ?

Решение

Шаг 1. Где границы бит. Каждый бит длится 10 мс, первый начался в 0. Значит, кусочки такие: 0–10 мс, 10–20 мс, ..., 70–80 мс. На рисунке их уже отделяет пунктир – восемь кусочков, как и бит.

Шаг 2. Какой размах считать громким. В записи встречаются колебания двух размеров: до 1 и до 0,25. Большой размах – громко, маленький – тихо. Порог можно поставить посередине, около 0,6: всё, что выше, – единица.

Шаг 3. Читаем по кусочкам.

Таблица 5.2

Кусочек	Время, мс	Размах	Бит
1	0–10	1	1
2	10–20	0,25	0
3	20–30	0,25	0
4	30–40	1	1
5	40–50	1	1
6	50–60	0,25	0
7	60–70	1	1
8	70–80	0,25	0

Шаг 4. Сколько колебаний в одном бите. Для проверки: $\tau = 10$ мс = 0,01 с, $n = 1000 \cdot 0,01 = 10$. Посчитай вершины в первом кусочке – их 10. Рисунок и условие друг другу не противоречат, границы бит мы провели правильно.

 **Типичная ошибка.** Записать тихие кусочки единицами: «там же есть колебания, значит, что-то передаётся». В этой задаче тихо – это ноль. Тишина вместо тихого тона бывает в других задачах, и там ноль узнать ещё проще. Вторая ошибка – прочитать 30–50 мс как одну единицу, потому что громкая пачка сплошная. Она длится 20 мс, это два бита.

Ответ: 10011010.

Своя задача по программе сферы, модуль 4

Реши сам

5.1. Приёмник измерил огибающую в середине каждого из восьми бит: 64, 70, 12, 68, 15, 11, 66, 13. Громко – 1, тихо – 0, порог 40: всё, что 40 и больше, – единица. Запиши принятые биты одной строкой.

5.2. С дачи на крыльцо передают сообщение ИК-лучом: луч горит – 1, погас – 0, скорость 20 бит/с. Фотоприёмник записал: луч горел 0,15 с, не горел 0,1 с, горел 0,05 с, не горел 0,2 с, горел 0,1 с. Сколько бит пришло и какие? В ответ введи число бит и сами биты через пробел.

Задача 5.2 Команды для КиберМишки

амплитудная модуляция, тон есть или нет, длинные пачки, скорость передачи

КиберМишке передают команды звуком: тон есть – 1, тишина – 0, скорость 100 бит/с. Каждая команда – два бита:

Таблица 5.3

Биты	00	01	10	11
Команда	вперёд	назад	налево	направо

Запись с микрофона – на рисунке. Колебаний несущей так много, что на графике они слились в сплошные полосы. Передача началась в момент 0 и длилась 120 мс. Какие биты получил Мишка и сколько раз он повернёт направо? В ответ введи биты одной строкой и число поворотов направо через пробел.

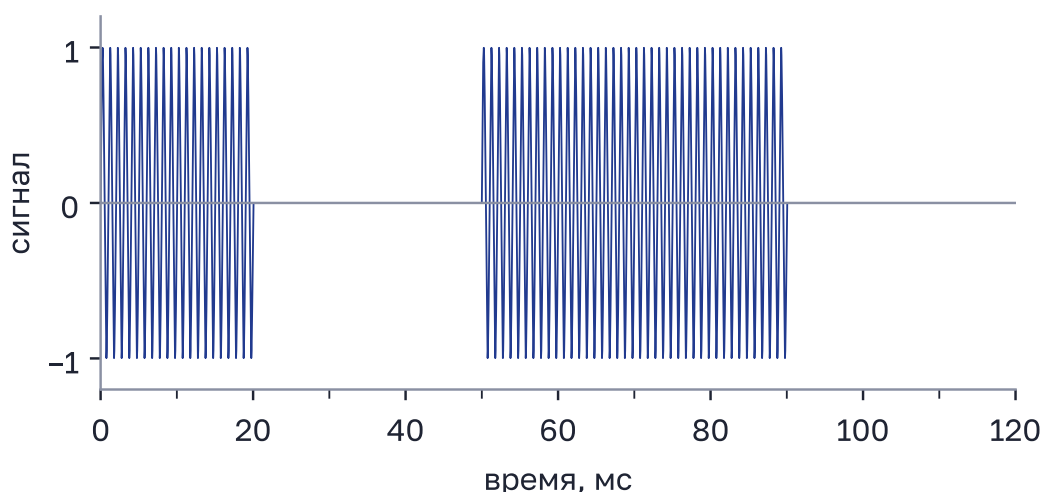


Рисунок 5.11 – Запись с микрофона: две пачки тона, между ними и после них тишина

Дано: тон – 1, тишина – 0; $v = 100$ бит/с; запись 0–120 мс; команды по 2 бита.

Найти: биты; сколько раз «направо».

Решение

Шаг 1. Длительность бита. $\tau = 1 : v = 1 : 100 = 0,01$ с. Переводим в миллисекунды: $0,01$ с = 10 мс. Проверим общую длину: $120 \text{ мс} : 10 \text{ мс} = 12$ бит – это как раз шесть команд по два бита.

Шаг 2. Пачки и паузы по графику. Читаем, где начинается и кончается каждая полоса. Маленькие засечки на оси – через 10 мс.

- 0–20 мс – тон, 20 мс;
- 20–50 мс – тишина, 30 мс;
- 50–90 мс – тон, 40 мс;

- 90–120 мс – тишина, 30 мс.

Шаг 3. Сколько бит в каждой пачке. Делим длительность на 10 мс:

- тон 20 мс $\rightarrow 20:10 = 2$ единицы: 11;
- тишина 30 мс $\rightarrow 3$ нуля: 000;
- тон 40 мс $\rightarrow 4$ единицы: 1111;
- тишина 30 мс $\rightarrow 3$ нуля: 000.

Вместе: 11 000 1111 000 $\rightarrow$ **110001111000**. Бит $2 + 3 + 4 + 3 = 12$, как и насчитали в шаге 1.

Шаг 4. Режим на команды по два бита. 11 | 00 | 01 | 11 | 10 | 00 $\rightarrow$ направо, вперёд, назад, направо, налево, вперёд. Направо Мишка повернёт 2 раза.

Проверим кодом. Строки после `#` – пояснения, Python их не читает.

```
pachki = [("1", 20), ("0", 30), ("1", 40), ("0", 30)] # (что звучало, сколько мс)
tau = 1000 / 100 # длительность бита в мс: 10.0
bity = ""
for znak, dlit in pachki: # берём пачки по очереди
    bity = bity + znak * round(dlit / tau) # "1" * 2 = "11"
print(bity) # 110001111000
komandy = {"00": "вперёд", "01": "назад", "10": "налево", "11": "направо"}
napravo = 0
for i in range(0, len(bity), 2): # i = 0, 2, 4, ... - начало каждой команды
    kom = komandy[bity[i:i + 2]] # два бита  $\rightarrow$  слово из таблицы
    print(kom)
    if kom == "направо":
        napravo = napravo + 1
print(napravo) # 2
```

В первой строке – список пачек: каждая пачка записана парой «какой бит – сколько миллисекунд». Первый цикл `for` проходит по пачкам, делит длительность на τ и столько раз повторяет символ: в Python `"1" * 2` даёт `"11"`. Словарь `komandy` хранит таблицу из условия: по двум битам он выдаёт слово. Второй цикл шагает по строке через два символа, `range(0, len(bity), 2)` даёт номера 0, 2, 4 и так до 10. Запись `bity[i:i + 2]` вырезает два символа, начиная с номера `i`. Если команда оказалась «направо», счётчик `napravo` растёт на один.

⚠ Типичная ошибка. Считать каждую пачку одним битом и записать 1010. Получится 4 бита вместо 12, и команд выйдут две вместо шести. Приёмник не видит границ между одинаковыми битами, он знает только длительность пачки. Вторая ловушка – прочитать край второй пачки как 80 или 100 мс. Он стоит посередине между подписями 80 и 100, у засечки 90.

Ответ: 110001111000 2.

Своя задача по программе сферы, модуль 9

Реши сам

5.3. Задача наоборот. Нужно передать КиберМишке команды «налево, вперёд, направо» по таблице из задачи 5.2: тон есть – 1, тишина – 0, скорость 50 бит/с. Сколько миллисекунд всего будет звучать тон и сколько длится самая длинная тишина внутри передачи? В ответ введи два числа через пробел.

5.4. В кабинете шумно. Когда передатчик молчит, приёмник всё равно показывает от 8 до 22, а когда звучит тон – от 55 до 70. Пришли значения в середине восьми бит: 58, 21, 64, 12, 18, 70, 55, 9. Какой порог из трёх – 20, 30 или 60 – прочитает сообщение без ошибок? Всё, что не меньше порога, – единица. В ответ введи порог и биты через пробел.

Задача 5.3 Высокий и низкий

частотная модуляция, частота по числу колебаний, перевод мс в с

Передатчик отправил 6 бит частотной модуляцией. Какой частотой передаётся единица, а какой ноль, в условии не сказано. Известно только, что частоты две, единица – более высокий тон, а каждый бит длится 10 мс. Запись – на рисунке. Найди частоту нуля, частоту единицы и само сообщение. В ответ введи через пробел частоту нуля в герцах, частоту единицы в герцах и 6 бит.

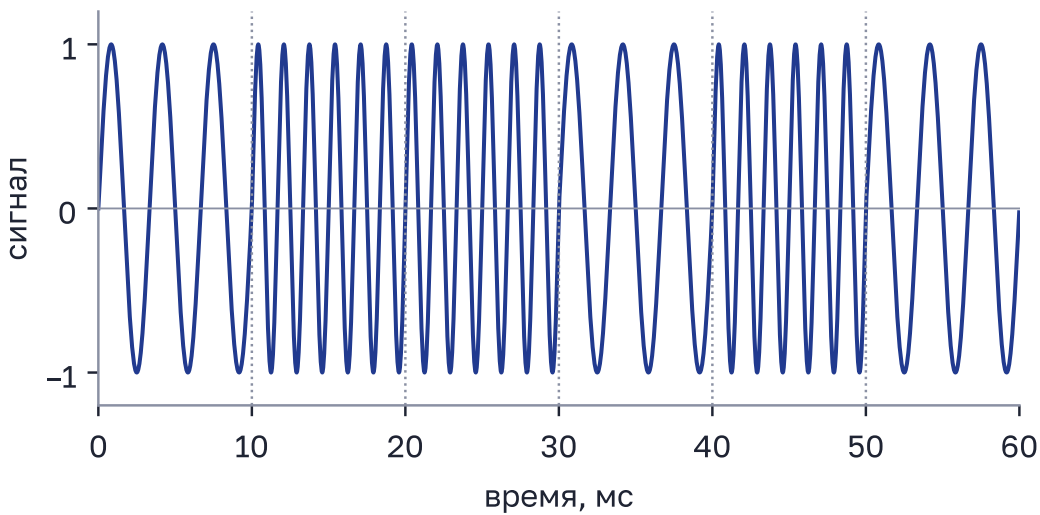


Рисунок 5.12 – Запись с микрофона. Пунктир – границы бит

Дано: ЧМ, 6 бит; $\tau = 10$ мс; выше тон – 1.

Найти: f_0, f_1 – ?; биты – ?

Решение

Шаг 1. Считаем колебания в кусочках. Одно колебание – одна верхушка. Идём по кусочкам между пунктирами:

Таблица 5.4

Кусочек, мс	Верхушек
0–10	3

Кусочек, мс	Верхушек
10–20	6
20–30	6
30–40	3
40–50	6
50–60	3

Встречаются только два числа, 3 и 6, – значит, частот в сообщении и правда две.

Шаг 2. Переводим миллисекунды в секунды. $\tau = 10 \text{ мс} = 10:1000 = 0,01 \text{ с}$.

Шаг 3. Частоты. $f = n: \tau$. Три колебания: $3:0,01 = 300 \text{ Гц}$. Шесть колебаний: $6:0,01 = 600 \text{ Гц}$.

Шаг 4. Какая частота – единица. По условию единица – более высокий тон, то есть 600 Гц. Ноль – 300 Гц.

Шаг 5. Биты. $3 \rightarrow 0, 6 \rightarrow 1, 6 \rightarrow 1, 3 \rightarrow 0, 6 \rightarrow 1, 3 \rightarrow 0$. Сообщение **011010**.

Обе частоты человек слышит без труда. 600 Гц вдвое больше 300 Гц – музыкант сказал бы, что тоны отличаются на октаву, как «до» и следующее «до» на пианино.

 **Типичная ошибка.** Разделить число колебаний на 10, не переведя миллисекунды: $6:10 = 0,6 \text{ Гц}$. Такой медленный «звук» – одно колебание почти за две секунды – человек не услышит, и это сразу выдаёт ошибку. Ещё одна ловушка – считать и верхушки, и впадины: выйдет 6 и 12, а частоты удвоятся до 600 и 1200 Гц.

Ответ: 300 600 011010.

Своя задача по программе сферы, модуль 4

Реши сам

5.5. ЧМ-передатчик: единица – 2400 Гц, ноль – 1200 Гц, бит длится 5 мс. Сколько колебаний поместится в бите с единицей и в бите с нулём? В ответ введи два числа через пробел.

5.6. Приёмник не считает колебания, а измеряет период в каждом бите. Получилось по порядку: 0,25 мс; 0,5 мс; 0,5 мс; 0,25 мс; 0,25 мс; 0,5 мс; 0,25 мс. Единица – более высокий тон. Найди две частоты передатчика и сообщение. В ответ введи через пробел меньшую частоту, большую частоту (в герцах) и биты.

Задача 5.4 Выбираем частоты

выбор частот, спектр шума, целое число колебаний

Команда готовит ЧМ-передачу звуком в кабинете. Условия такие.

- Динамик и микрофон хорошо работают от 500 до 5000 Гц.
- Спектр шума в кабинете – на рисунке. Частота передачи должна отстоять от каждого пика шума не меньше чем на 200 Гц.
- Бит длится 10 мс. В бите должно помещаться целое число колебаний, и не меньше 10.

Из списка частот 1) 450 Гц; 2) 1250 Гц; 3) 1800 Гц; 4) 2900 Гц; 5) 3600 Гц; 6) 4200 Гц; 7) 700 Гц; 8) 5500 Гц: а) выбери все, которые подходят; б) из подходящих выбери пару для нуля и единицы с самой большой разницей частот. В ответ введи через пробел два числа: номера из пункта а) по возрастанию, слитно, и два номера пары из пункта б) по возрастанию, тоже слитно. Например, если бы подошли номера 1, 2 и 7, а парой были 1 и 7, ответ был бы «127 17».

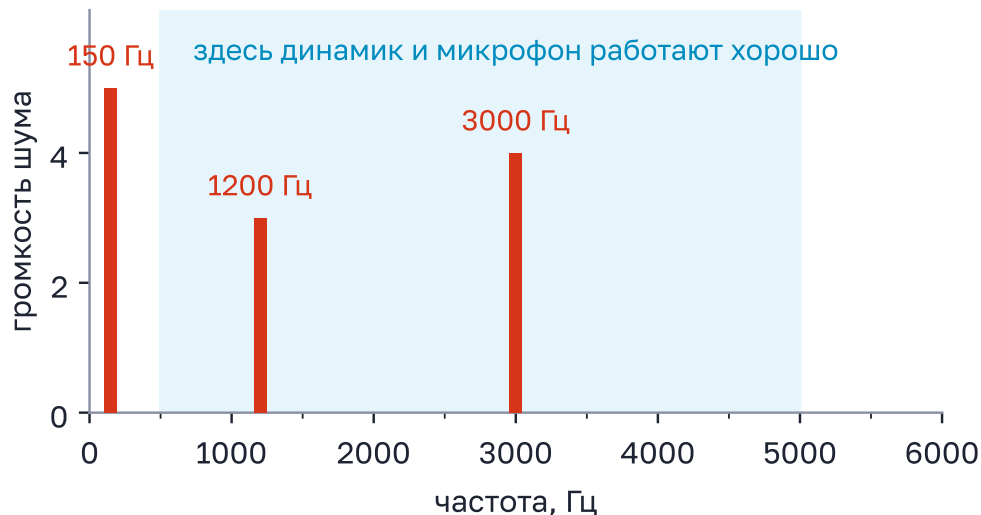


Рисунок 5.13 – Спектр шума в кабинете: три пика. Светлая полоса – рабочий диапазон динамика и микрофона

Дано: диапазон 500–5000 Гц; пики шума 150, 1200, 3000 Гц; отступ от шума ≥ 200 Гц;
 $\tau = 10$ мс; $n = f \cdot \tau$ – целое, $n \geq 10$.

Найти: подходящие частоты; пару с наибольшей разницей.

Решение

Шаг 1. Диапазон. Отбрасываем то, что ниже 500 Гц или выше 5000 Гц: 1) 450 Гц и 8) 5500 Гц. Остаются 2, 3, 4, 5, 6, 7.

Шаг 2. Колебаний в бите. $\tau = 10$ мс = 0,01 с, $n = f \cdot 0,01$, то есть частоту делим на 100:

- 2. 1250 Гц $\rightarrow 12,5$ – не целое, не подходит;
- 3. 1800 Гц $\rightarrow 18$;
- 4. 2900 Гц $\rightarrow 29$;
- 5. 3600 Гц $\rightarrow 36$;
- 6. 4200 Гц $\rightarrow 42$;
- 7. 700 Гц $\rightarrow 7$ – меньше 10, не подходит.

Остаются 3, 4, 5, 6.

Шаг 3. Шум. Для каждой частоты находим расстояние до ближайшего пика шума. Пик 150 Гц лежит вне диапазона, но проверить его не вредно.

- 3. 1800 Гц: до 1200 – 600 Гц, до 3000 – 1200 Гц. Подходит.
- 4. 2900 Гц: до 3000 – всего 100 Гц. Не подходит – шум её заглушит.

- 5. 3600 Гц: до 3000 – 600 Гц. Подходит.
- 6. 4200 Гц: до 3000 – 1200 Гц. Подходит.

Ответ пункта а): 3, 5, 6.

Шаг 4. Пара с наибольшей разницей. Возможные пары: 1800 и 3600 – разница 1800 Гц; 1800 и 4200 – 2400 Гц; 3600 и 4200 – 600 Гц. Больше всего у пары 3 и 6. Номера пары слитно по возрастанию – 36.

1800 и 4200 Гц отличаются больше чем вдвое, на слух это совсем разные пiski – приёмник их точно не спутает.

 **Типичная ошибка.** Проверить только одно правило и остановиться. Частота 2900 Гц проходит и по диапазону, и по колебаниям (29 – целое), поэтому её часто берут. Но она стоит вплотную к пику шума 3000 Гц. Условий три – проверять нужно все три для каждой частоты.

Ответ: 356 36.

Своя задача по программе сферы, модуль 11

Реши сам

5.7. Другой кабинет: динамик и микрофон хорошо работают от 300 до 3000 Гц, пики шума на 1000 и 2000 Гц, отступ от шума – не меньше 150 Гц. Бит длится 20 мс, в бите должно помещаться целое число колебаний, не меньше 20. Какие из частот подходят: 1) 800 Гц; 2) 1300 Гц; 3) 1525 Гц; 4) 1950 Гц; 5) 2600 Гц; 6) 3400 Гц; 7) 1600 Гц? В ответ введи номера по возрастанию без пробелов.

5.8.* Частоты уже выбраны: единица – 1500 Гц, ноль – 2500 Гц. Какую самую короткую длительность бита (целое число миллисекунд) можно взять, чтобы у обеих частот в бите помещалось целое число колебаний, и у каждой – не меньше 10? В ответ введи длительность в миллисекундах.

Задача 5.5 Сколько ждать сообщения

время передачи, скорость, служебные биты, промежутки

Метеостанция на школьном дворе передаёт по ИК-каналу сводку из 20 символов. Каждый символ – 8 бит, и после каждого символа идёт бит чётности. Перед сводкой – стартовая метка из 8 бит. Между соседними символами – пауза длиной в 2 бита, перед первым символом и после последнего паузы нет. Скорость передачи – 200 бит/с. Сколько секунд идёт вся передача? Ответ округли до сотых.

Дано: 20 символов по $8 + 1$ бит; старт 8 бит; паузы по 2 бита между символами; $v = 200$ бит/с.

Найти: t , с – ?

Решение

Шаг 1. Из чего состоит передача. Выписываем столбиком:

- стартовая метка – 8 бит;
- символы: каждый 8 бит данных + 1 бит чётности = 9 бит, всего $20 \cdot 9 = 180$ бит;
- паузы: между 20 символами $20 - 1 = 19$ промежутков, по 2 бита – $19 \cdot 2 = 38$ бит.

Шаг 2. Всего бит. $I = 8 + 180 + 38 = 226$ бит.

Шаг 3. Время. $t = I : v = 226 : 200 = 1,13$ с.

Тот же ответ через длительность бита: $\tau = 1 : 200 = 0,005$ с, $t = 226 \cdot 0,005 = 1,13$ с.

Чуть больше секунды на двадцать символов – примерно столько же занимает сказать вслух «плюс три, ветер слабый». Медленно для интернета, но для метеостанции во дворе хватает с запасом.

 **Типичная ошибка.** Посчитать 20 пауз вместо 19: получится 228 бит и 1,14 с. Это ошибка забора – между 20 столбиками 19 пролётов. Не реже забывают стартовую метку или бит чётности: без метки выходит 1,09 с, без чётности – 1,03 с. Поэтому и выписываем все части столбиком до того, как делить.

Ответ: 1,13 с.

Своя задача по программе сферы, модуль 11

Реши сам

5.9. КиберМишке передают 150 бит, каждый бит длится 20 мс. Сколько секунд займёт передача?

5.10. Канал передаёт 50 бит/с. В начале идёт стартовая метка из 8 бит, потом команды по 4 бита, причём каждый бит команды для надёжности повторяют три раза. Сколько целых команд успеет пройти за 10 с?

Задача 5.6 Какой датчик врёт

надёжность канала, сравнение рядов, доля ошибок в процентах

В теплице на даче четыре датчика влажности передают показания по ИК-каналу. Показания стали странными, и хозяйка проверила связь: отправила каждому датчику по 20 бит, а датчик вернул то, что услышал.

Таблица 5.5

Датчик	Отправлено	Вернулось
1	10110010011101001011	10111010011101001011
2	01101110100011010110	00001110100011010100

Датчик	Отправлено	Вернулось
3	11100101001101110001	11100101001101110001
4	00111010110010011101	00111011100010011101

У какого датчика самая ненадёжная связь и какая у него доля ошибок? Какая доля ошибок по всем четырём датчикам вместе? В ответ введи через пробел номер датчика, его долю ошибок в процентах и общую долю ошибок в процентах.

Дано: 4 датчика; по 20 отправленных и 20 вернувшихся бит.

Найти: худший датчик; его доля ошибок, %; общая доля ошибок, %.

Решение

Шаг 1. Сравниваем бит за битом. Удобно разбить строки на четвёрки и сравнивать их. Датчик 1: 1011 0010 0111 0100 1011 и 1011 1010 0111 0100 1011 – отличается одна четвёрка, 0010 и 1010, в ней один бит. Ошибка одна, на 5-м месте.

Так же для остальных:

- датчик 2: 0110 1110 1000 1101 0110 и 0000 1110 1000 1101 0100 – ошибки на 2-м, 3-м и 19-м местах, всего 3;
- датчик 3: строки совпадают, ошибок 0;
- датчик 4: 0011 1010 1100 1001 1101 и 0011 1011 1000 1001 1101 – ошибки на 8-м и 10-м местах, всего 2.

Шаг 2. Худший датчик. Больше всего ошибок у датчика 2: 3 из 20.

Шаг 3. Его доля ошибок. $3:20 \cdot 100 = 15\%$.

Шаг 4. Общая доля. Ошибок всего $1 + 3 + 0 + 2 = 6$. Бит отправлено $4 \cdot 20 = 80$. Доля $6:80 \cdot 100 = 7,5\%$.

Пятнадцать процентов – это примерно каждый седьмой бит. С такой связью датчик влажности легко «покажет» засуху после полива, так что начинать ремонт стоит со второго.

Сравнение бит за битом – работа как раз для компьютера.

```
otpr = ["10110010011101001011", "01101110100011010110",
        "11100101001101110001", "00111010110010011101"]
vern = ["10111010011101001011", "00001110100011010100",
        "11100101001101110001", "00111011100010011101"]
vsego = 0
for nomer in range(4):                # датчики 0, 1, 2, 3
    oshibki = 0
    for k in range(20):                # места 0 ... 19 в строке
        if otpr[nomer][k] != vern[nomer][k]:
            oshibki = oshibki + 1
    vsego = vsego + oshibki
    print(nomer + 1, oshibki, oshibki / 20 * 100) # номер, ошибки, %
print(vsego / 80 * 100)                 # 7.5
```

`otpr[nomer]` – строка, отправленная датчику с этим номером, а `otpr[nomer][k]` – символ на месте `k` в этой строке. Знак `!=` значит «не равно». Внутренний цикл проходит все 20 мест и

прибавляет единицу к `oshibki` за каждое несовпадение. Python считает с нуля, поэтому при печати к номеру датчика добавляем 1.

❗ Типичная ошибка. В шаге 3 делить на все 80 бит: $3:80 \cdot 100 = 3,75\%$. Доля ошибок у одного датчика считается от того, что отправили ему, – от 20 бит. На все 80 делят только в шаге 4, где ошибки тоже собраны со всех датчиков. Ещё бывает, что при сравнении длинных строк глаз «съезжает» на соседний бит – поэтому режь на четвёрки.

Ответ: 2 15 7,5.

По мотивам задачи первого отборочного этапа НТО 2025/26, профиль «Технологии беспроводной связи», «Девять из десяти прогнозов не ошибаются»

Реши сам

5.11. По трём каналам отправили по 12 бит. Канал 1: отправлено `101100111000`, пришло `101110111010`. Канал 2: отправлено `011011010010`, пришло `011011010010`. Канал 3: отправлено `110001011101`, пришло `110001001101`. У какого канала больше всего ошибок и сколько их? В ответ введи номер канала и число ошибок через пробел.

5.12. Чтобы датчик из задачи 5.6 не врал, каждый его бит стали передавать трижды, а приёмник голосует большинством. Пришло 24 бита, по тройкам: `111 010 110 000 101 001 111 011`. Считай, что в каждой тройке испорчено не больше одного бита. Какие 8 бит отправил датчик и в скольких тройках помеха что-то испортила? В ответ введи биты и число таких троек через пробел.

Задача 5.7* Два передатчика в одной комнате

два передатчика, разделение по частоте, порог для каждого канала, время передачи

В кабинете одновременно работают два передатчика. Передатчик А шлёт 8 бит на несущей 1000 Гц, передатчик Б – 8 бит на несущей 3000 Гц. Оба используют АМ: тон есть – 1, тишина – 0, бит длится 50 мс. Микрофон слышит оба сразу. Программа приёмника режет запись на кусочки по 50 мс и для каждого строит спектр. В таблице – высота столбика спектра на 1000 Гц и на 3000 Гц в каждом кусочке.

Таблица 5.6

Кусочек	1000 Гц	3000 Гц
1	41	4
2	3	27
3	39	30
4	44	2
5	2	25

Кусочек	1000 Гц	3000 Гц
6	5	28
7	40	3
8	38	1

а) Какое сообщение передал А, какое – Б? б) Сколько секунд длилась передача обоих сообщений? в) Сколько секунд заняла бы передача, если бы передатчики работали по очереди? В ответ введи через пробел сообщение А, сообщение Б и ответы б) и в).

Дано: А – 1000 Гц, Б – 3000 Гц; АМ «тон есть – нет»; $t = 50$ мс; по 8 бит; таблица спектра.

Найти: сообщения А и Б; время одновременной передачи и передачи по очереди.

Решение

Шаг 1. Разделяем передатчики. Звуки смешались в воздухе, но спектр раскладывает их обратно: столбик на 1000 Гц показывает только передатчик А, столбик на 3000 Гц – только Б. Это два отдельных канала, и читать их будем отдельно.

Шаг 2. Порог для канала А. Выпишем числа столбца 1000 Гц по возрастанию: 2, 3, 5 | 38, 39, 40, 41, 44. Они чётко делятся на две группы – «тишина» около 2–5 и «тон» около 38–44. Порог ставим посередине между группами, например 20: между 5 и 38 любой порог сработает.

Сообщение А: $41 \rightarrow 1$, $3 \rightarrow 0$, $39 \rightarrow 1$, $44 \rightarrow 1$, $2 \rightarrow 0$, $5 \rightarrow 0$, $40 \rightarrow 1$, $38 \rightarrow 1$. Получается **10110011**.

Шаг 3. Порог для канала Б. Столбец 3000 Гц по возрастанию: 1, 2, 3, 4 | 25, 27, 28, 30. Здесь «тон» слабее – передатчик Б, видимо, стоит дальше от микрофона. Порог опять посередине, например 14.

Сообщение Б: $4 \rightarrow 0$, $27 \rightarrow 1$, $30 \rightarrow 1$, $2 \rightarrow 0$, $25 \rightarrow 1$, $28 \rightarrow 1$, $3 \rightarrow 0$, $1 \rightarrow 0$. Получается **01101100**.

Шаг 4. Время одновременной передачи. Передатчики работали одновременно, каждый 8 бит по 50 мс. $50 \text{ мс} = 0,05 \text{ с}$, $t = 8 \cdot 0,05 = 0,4 \text{ с}$.

Шаг 5. Время по очереди. Сначала 8 бит одного, потом 8 бит другого – всего 16 бит: $16 \cdot 0,05 = 0,8 \text{ с}$. Одновременная передача на разных частотах вдвое быстрее.

Для КиберМишки это значит, что две команды приходят за время одной – и никто в очереди не стоит.

⚠ Типичная ошибка. Взять один порог на оба канала. С порогом 30 канал А читается верно, а в канале Б единицами окажется только кусочек 3 (там 30), и сообщение Б выйдет 00100000. Порог для каждого канала ищи по разрыву между группами чисел в его собственном столбце.

Ответ: 10110011 01101100 0,4 0,8.

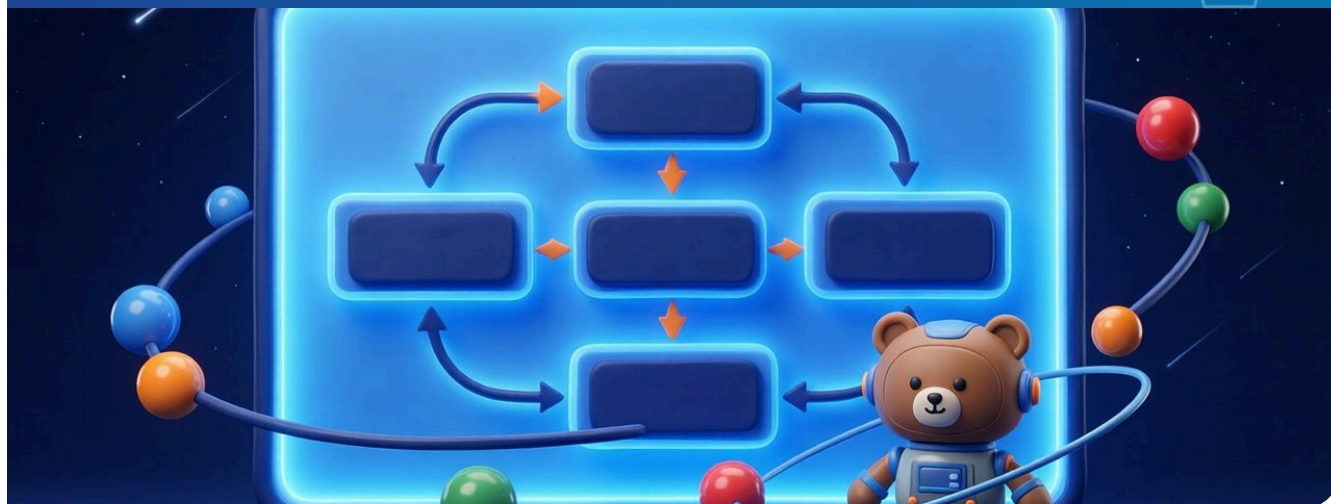
По мотивам задачи заключительного этапа НТО 2025/26, профиль «Технологии беспроводной связи», «Полигон-А: деконволюция»

Реши сам

5.13.* Два передатчика работают на 1500 Гц и 2500 Гц, АМ «тон есть – нет», по 6 бит. Высота столбика спектра в шести кусочках: на 1500 Гц – 33, 31, 2, 4, 35, 1; на 2500 Гц – 3, 12, 14, 2, 1, 13. Какие сообщения передали? В ответ введи через пробел сообщение на 1500 Гц и сообщение на 2500 Гц.

5.14.* КиберМишка стоит в конце спортзала, в 34 м от передатчиков. Нужно передать ему 120 бит. ИК-канал работает со скоростью 40 бит/с, звуковой – 20 бит/с. Скорость звука – 340 м/с, задержкой ИК-луча можно пренебречь. а) Сколько секунд займёт передача, если всё отправить по ИК? б) Сообщение поделили: 80 бит пошли по ИК, 40 – звуком, оба передатчика начали одновременно. Через сколько секунд у Мишки будут все 120 бит? В ответ введи два числа через пробел.

ПРИМС: машины состояний в Кибериаде



Программу для КиберМишки в этой сфере не пишут строчками кода. Её рисуют: прямоугольники – режимы работы, стрелки – что их переключает. Такая схема называется машиной состояний, и в этой главе ты научишься её читать, проверять и дописывать. Задачи по этой теме похожи на головоломки. Дана схема и цепочка событий – где окажется устройство? Где ошибка в чужой схеме? Какой стрелки не хватает? В ответ пишут название или номер состояния, число или строку из цифр. Оцениваемые задания курса по ПРИМС делают в Кибериаде на компьютере, а тетрадь и аккуратная табличка нужны для подготовки: на бумаге схему сначала продумывают и проверяют.

6.1. Состояние: устройство помнит, что с ним было

НА ПАЛЬЦАХ

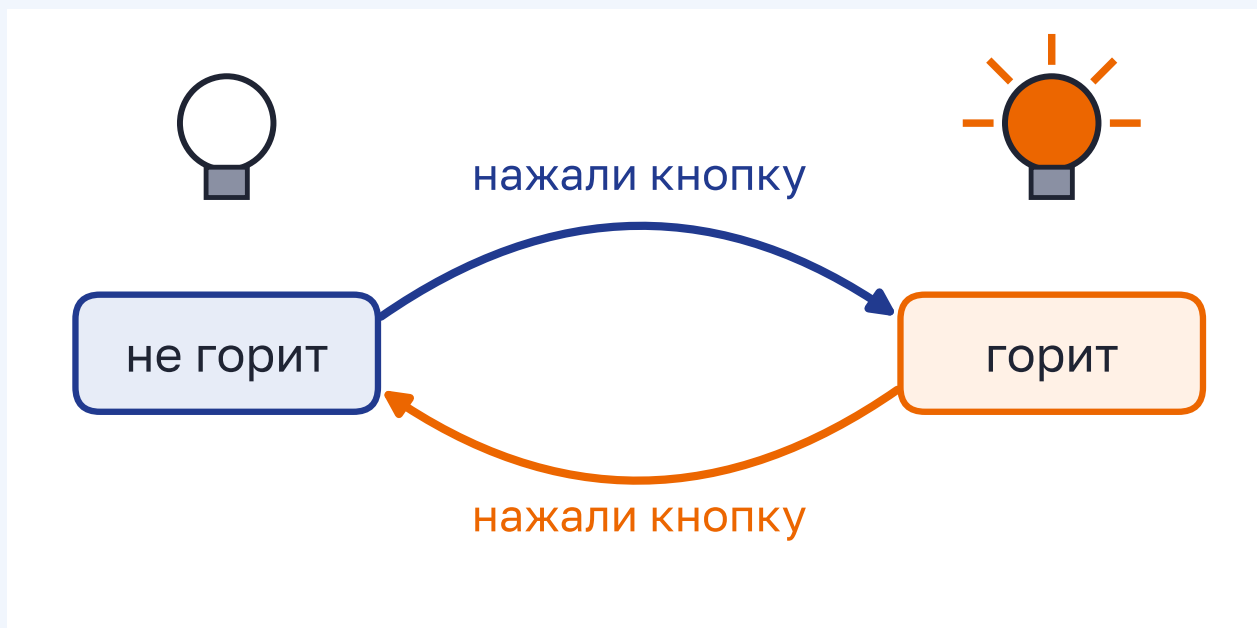


Рисунок 6.1 – Настольная лампа с одной кнопкой

Кнопка одна и та же, а результат разный: лампа помнит, горит она сейчас или нет.

У настольной лампы одна кнопка. Нажал – загорелась, нажал ещё раз – погасла. Откуда лампа знает, что сделать? Нажатие каждый раз одинаковое. Всё решает то, в каком **состоянии** лампа была до нажатия: «горит» или «не горит».

Состояние – это режим, в котором устройство находится прямо сейчас. У светофора три состояния: красный, жёлтый, зелёный. У телефона – «заблокирован» и «разблокирован». У стиральной машины – «стирка», «полоскание», «отжим». Пока ничего не случилось, устройство остаётся в своём состоянии сколько угодно: светофор горит красным, лампа горит.

Устройство, которое так себя ведёт, называют **машиной состояний**. У неё есть:

- список состояний, их конечное число – три, пять, десять, но не бесконечно много;
- **начальное состояние** – то, в котором машина оказывается сразу после включения;
- правила, по которым одно состояние сменяется другим.

В каждый момент машина находится **в одном** состоянии. Лампа не может гореть и не гореть сразу, светофор не горит красным и зелёным одновременно.

Правила удобнее всего нарисовать. Такой рисунок называется **диаграммой состояний**. В нашей книге состояние – это прямоугольник со скруглёнными углами, внутри – название. Переход из состояния в состояние – стрелка. Начальное состояние отмечено чёрной точкой, от которой к нему идёт короткая стрелка.

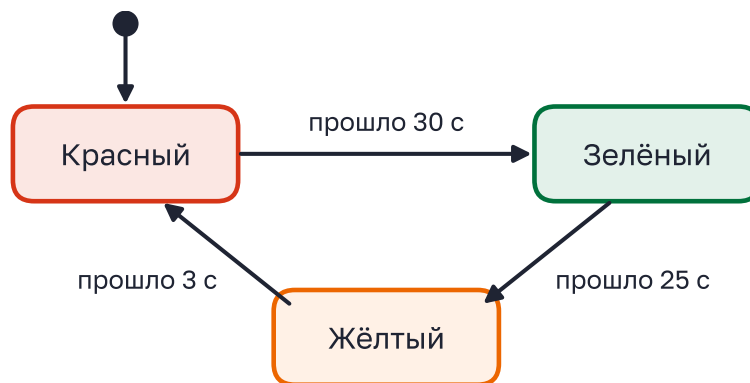


Рисунок 6.2 – Диаграмма состояний светофора для машин. Светофор включается в красном

Прочитаем схему светофора. Светофор включили – он в состоянии «Красный». Через 30 с он переходит в «Зелёный», ещё через 25 с – в «Жёлтый», ещё через 3 с – снова в «Красный». Дальше всё повторяется. Один круг длится $30 + 25 + 3 = 58$ с.

Какой сигнал горит через 100 с после включения? Отбросим целый круг: $100 - 58 = 42$ с. Значит, светофор на 42-й секунде второго круга. Красный горит с 0-й по 30-ю секунду круга, зелёный – с 30-й по 55-ю. 42 попадает в зелёный промежуток – горит зелёный.

Названия состояний выбирают так, чтобы они отвечали на вопрос «что сейчас происходит?»: «жду», «слушаю», «еду вперёд». «Состояние 1» и «Режим Б» – плохие имена, через неделю сам не вспомнишь, что они значат.

Запомни. Состояние – это режим, в котором устройство сейчас. Машина всегда в одном состоянии и остаётся в нём, пока её не переключат. Чёрная точка показывает, откуда машина начинается.

6.2. Событие, условие, действие, переход

НА ПАЛЬЦАХ

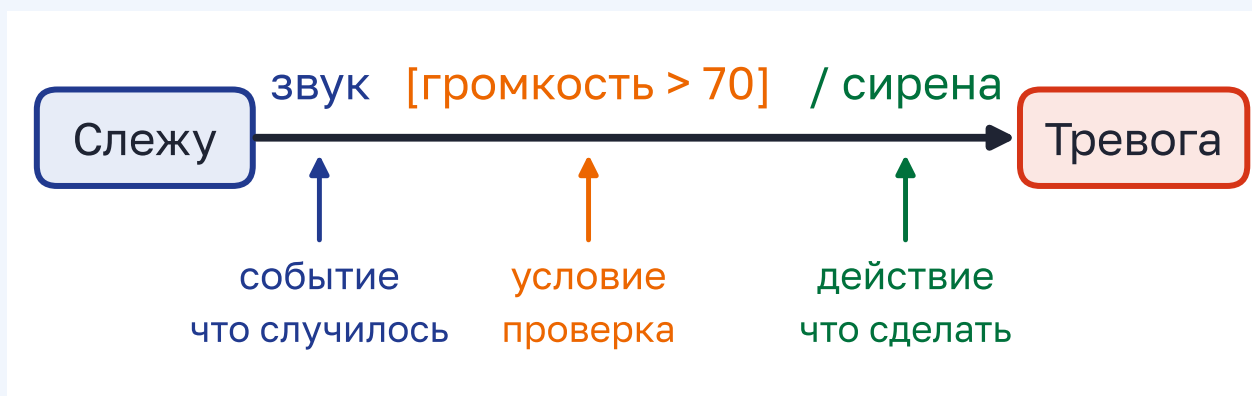


Рисунок 6.3 – Подпись на стрелке из трёх частей

Случился звук. Проверили: громкость больше 70? Если да – включили сирену и перешли в «Тревогу».

Что заставляет машину сменить состояние? **Событие** – то, что случилось в какой-то момент. Нажали кнопку. Датчик услышал хлопок. Прошло 30 секунд. Пришёл сигнал по инфракрасному лучу. Событие длится мгновение: хлопнули – и всё, хлопок кончился.

Переход – смена одного состояния другим. На диаграмме это стрелка, и на ней подписано, по какому событию она срабатывает. Если случилось событие, машина смотрит только на стрелки, которые выходят **из текущего состояния**. Нашлась стрелка с этим событием – машина переходит по ней. Не нашлась – событие пропадает впустую, машина остаётся, где была. Лампа с тремя режимами (она ждёт тебя в задаче 6.1) в выключенном состоянии не отзывается на долгое нажатие: такой стрелки у неё нет.

Условие – проверка «да или нет», которая стоит на стрелке в квадратных скобках. Когда случилось событие, машина проверяет условие. Верно – переход происходит. Неверно – стрелка не срабатывает, будто её нет. Условие позволяет одному событию вести себя по-разному: хлопок тихий – ничего, хлопок громкий – тревога.

Действие – то, что устройство делает: включить сирену, показать картинку на экранчике, отправить сообщение, прибавить 1 к счётчику. На стрелке действие пишут после косой черты. Оно выполняется, когда машина проходит по этой стрелке. Бывают и действия, привязанные к самому состоянию: действие **на входе** выполняется один раз, когда машина попала в состояние, действие **на выходе** – один раз, когда она из него уходит.

Полная подпись стрелки читается так:

событие [условие] / действие.

Условие и действие можно не писать, если их нет. Тогда на стрелке останется одно событие, как у светофора: «прошло 30 с».

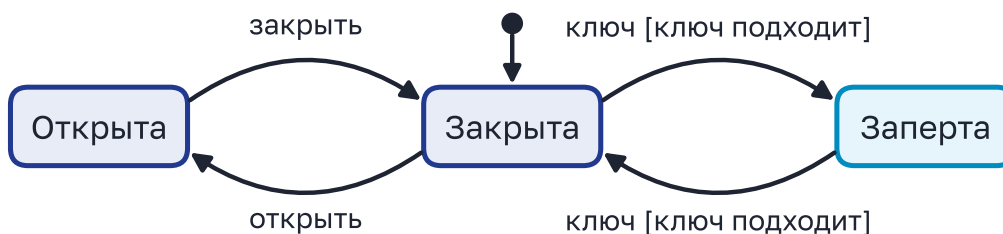


Рисунок 6.4 – Дверь с замком. Ключом можно запереть только закрытую дверь

Разберём дверь в кабинет информатики. Три состояния: «Открыта», «Закрыта», «Заперта». Сначала дверь закрыта, но не заперта. Из «Закрыта» можно открыть дверь – и она «Открыта». Можно повернуть ключ – и она «Заперта», но только если ключ подходит: условие [ключ подходит]. Чужой ключ в замке проворачиваться не будет, стрелка не сработает, дверь останется закрытой. Из «Заперта» тем же ключом дверь отпирают обратно в «Закрыта».

Посмотри, чего на схеме нет. Из «Открыта» нет стрелки по событию «ключ». Повернёшь ключ в открытой двери – язычок замка выскочит, но дверь-то распахнута, запертой она от этого не станет. Из «Заперта» нет стрелки «открыть»: дёргай ручку сколько хочешь. Отсутствие стрелки – тоже правило. Оно говорит: в этом состоянии такое событие ничего не меняет.

Иногда стрелка возвращается в то же состояние, из которого вышла. Это **петля**. Она нужна, когда состояние менять не надо, а действие выполнить надо. Например, КиберМишка в состоянии «Считаю» на каждый хлопок прибавляет 1 к счётчику и продолжает считать: «хлопок

/ счётчик + 1». **Счётчик** здесь – это **переменная**, ячейка памяти с числом, которое программа меняет по ходу работы. Её можно проверять в условии: «хлопок [счётчик = 3] / показать сердечко».

Запомни. Подпись стрелки: событие [условие] / действие. Событие ищут только среди стрелок из текущего состояния. Нет стрелки или условие неверно – машина стоит на месте. Действие выполняется при переходе.

Совет. Когда читаешь чужую схему, проговаривай каждую стрелку полной фразой: «Если я в «Закрыта» и повернули ключ, и ключ подходит, то дверь переходит в «Заперта»». Длинно, зато не перепутаешь, откуда стрелка выходит и куда входит.

6.3. Трассировка: идём за машиной шаг за шагом

НА ПАЛЬЦАХ



Рисунок 6.5 – Цепочка событий для лампы и кадры, по которым видно, где она оказалась

Берёшь события по одному, как карточки из колоды, и каждый раз двигаешь фишку по нужной стрелке.

Трассировка – это когда ты сам, без компьютера, проигрываешь работу машины: берёшь события по порядку и записываешь, в каком состоянии машина после каждого. Слово пришло от английского trace – «след». Ты идёшь по следу программы. Это главный навык главы: почти любая задача про машину состояний сводится к трассировке.

Делай это таблицей. Одна строка – одно событие:

- **было** – в каком состоянии машина перед событием;
- **событие** – что случилось;
- **стрелка** – есть ли из состояния «было» стрелка с этим событием и верно ли её условие;
- **стало** – куда машина перешла (или «то же»).

Проследим за дверью из раздела 6.2. Она в «Закрыта». События по порядку: открыть, повернуть свой ключ, закрыть, повернуть свой ключ, открыть, повернуть чужой ключ.

Таблица 6.1

Шаг	Было	Событие	Стрелка	Стало
-----	------	---------	---------	-------

Шаг	Было	Событие	Стрелка	Стало
1	Закрыта	открыть	есть	Открыта
2	Открыта	свой ключ	нет	Открыта
3	Открыта	закрыть	есть	Закрыта
4	Закрыта	свой ключ	есть, ключ подходит	Заперта
5	Заперта	открыть	нет	Заперта
6	Заперта	чужой ключ	есть, но ключ не подходит	Заперта

Дверь заперта. Два из шести событий (шаги 2 и 5) прошли впустую, а на шаге 6 стрелка есть, но условие её не пустило. Ради таких строк таблица и нужна: в уме их проскакивают.

События от таймера. «Прошло 30 с» – тоже событие. Время отсчитывают от момента, когда машина **вошла** в состояние. Ушла и вернулась – отсчёт начинается заново. В задачах с таймерами удобно вести в таблице ещё и столбец «время»: в какой секунде машина вошла в состояние и в какой выйдет по таймеру.

Здесь мы упрощаем. В Кибериаде время задаёт компонент «Таймер»: его запускают действием (чаще на входе в состояние), а событие приходит от него. У Садовника таймер срабатывает один раз («Время вышло») и запускается заново, у других исполнителей бывает иначе – смотри условие. В задачах этой главы считай, как написано в условии.

Круги. Если машина ходит по кругу, длинную цепочку не обязательно проигрывать до конца. Светофор из раздела 6.1 проходит круг за 58 с – значит, через 58, 116, 174 секунды он снова в начале круга, на красном. Лампа с тремя режимами на каждом третьем нажатии возвращается туда, откуда начала. Раздели число событий на длину круга, а остаток доиграй по схеме. Только сначала убедись, что все события в цепочке одинаковые и круг ни разу не рвётся.

Запомни. Трассировка: состояние – событие – есть ли стрелка из **этого** состояния – верно ли условие – новое состояние. Одна строка на событие, без прыжков через шаги. Таймер считает с момента входа в состояние.

Совет. Пиши в таблице и те события, которые ничего не изменили. Составители задач любят подкладывать в цепочку нажатие, на которое у текущего состояния нет стрелки. Пропустишь такую строку – собьёшься со счёта на всех следующих.

6.4. Вложенные состояния

НА ПАЛЬЦАХ

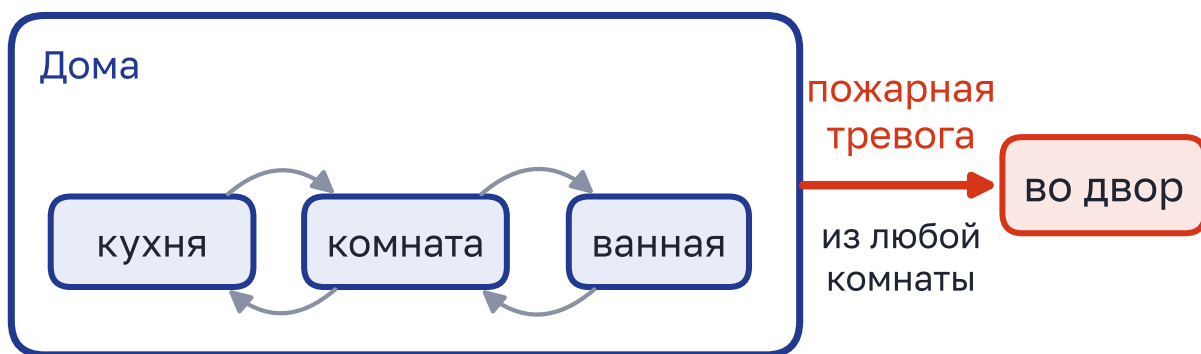


Рисунок 6.6 – Дом с комнатами и пожарная тревога

Ходишь по комнатам как хочешь. Но по пожарной тревоге выходишь во двор из любой комнаты – одно правило на весь дом.

Ты на кухне. Где ты? На кухне – и заодно дома. Из кухни можно пойти в комнату, из комнаты в ванную: это правила для комнат. А правило «при пожарной тревоге выходи во двор» пишут одно на весь дом, его не повторяют для каждой комнаты.

С машинами состояний так же. Состояние можно положить **внутри** другого. Внешнее называют **родительским**, внутренние – **вложенными** (или дочерними). На диаграмме родительское состояние – большая рамка, вложенные – прямоугольники внутри неё. Когда машина во вложенном состоянии, она заодно находится и в родительском.

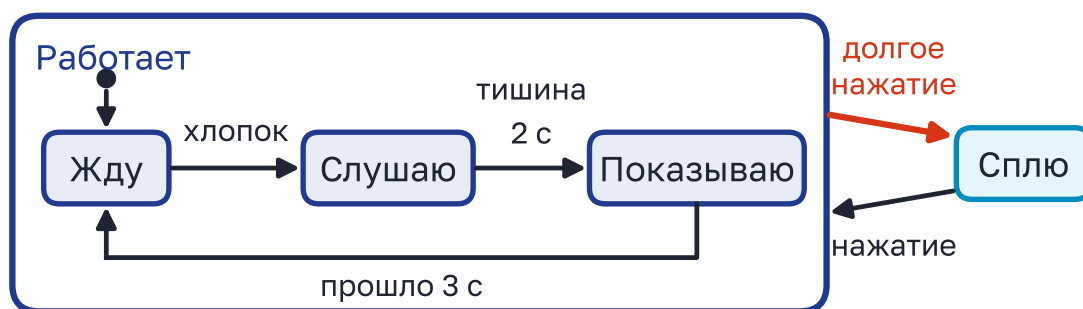


Рисунок 6.7 – КиберМишка: три рабочих состояния внутри «Работает» и сон снаружи

Вот КиберМишка, который слушает хлопки. После включения он в «Работает», а внутри рамки чёрная точка указывает на «Жду». Хлопок – Мишка переходит в «Слушаю». Две секунды тишины – в «Показываю»: выводит на экранчик, сколько насчитал. Через 3 с возвращается в «Жду». А долгое нажатие кнопки усыпляет Мишку, в каком бы из трёх рабочих состояний он ни был. Эта стрелка выходит из рамки «Работает».

Правила вложенных состояний:

- **Событие сначала проверяет самое внутреннее состояние.** Если у него есть стрелка с этим событием (и условие верно), срабатывает она.

- Если у вложенного состояния такой стрелки нет, событие передаётся родителю. Стрелка от рамки срабатывает из любого вложенного состояния.
- Стрелка, которая входит в рамку, ведёт в начальное вложенное состояние – туда, куда показывает чёрная точка внутри рамки. Мишка проснулся от нажатия – он снова в «Жду», хотя уснул, может быть, в «Показываю».
- Стрелка может вести и прямо к конкретному вложенному состоянию – тогда машина попадает в него.

Чем это выгодно? Без рамки стрелку «долгое нажатие» пришлось бы рисовать три раза – из «Жду», из «Слушаю» и из «Показываю». С рамкой она одна. Добавишь четвёртое рабочее состояние – оно само будет засыпать по долгому нажатию, и забыть эту стрелку не получится.

Теперь понятно и страшное название **ПРИМС** – «программирование расширенных иерархических машин состояний». **Иерархические** – потому что состояния вкладываются друг в друга, как комнаты в дом, а дом в город. **Расширенные** – потому что на схеме, кроме состояний и стрелок, записаны действия и условия. Так же расшифровывают это название в сборнике задач старшей НТО.

Запомни. Вложенное состояние первым отвечает на событие. Нет у него своей стрелки – отвечает рамка. Вошёл в рамку – попал в её начальное состояние. Одна стрелка от рамки заменяет по стрелке из каждого вложенного состояния.

6.5. Ошибки в диаграмме: тупики, недостижимые состояния, путаница

НА ПАЛЬЦАХ

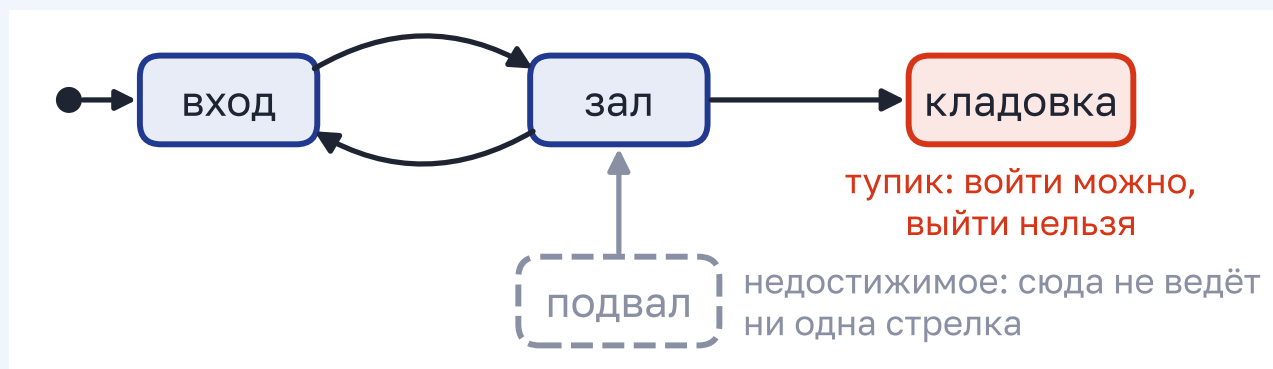


Рисунок 6.8 – План этажа: из кладовки нет выхода, в подвал нет входа

В кладовку войти можно, а выйти – нет. В подвал не ведёт ни одна дверь, и попасть туда невозможно.

Нарисовать схему легко. Нарисовать её без дыр – уже работа. Вот какие ошибки прячутся в чужих схемах – сначала заметные, потом хитрые.

Тупик – состояние, из которого не выходит ни одной стрелки. Машина попала туда – и застряла навсегда: никакое событие её не выведет, только выключить и включить заново. Ино-

гда так и задумано. Например, «Разряжен» у игрушки без зарядки – конец работы. Но если по условию устройство должно работать дальше, тупик – ошибка.

Недостижимое состояние – такое, в которое машина не может попасть из начального, какие бы события ни случались. Его программист нарисовал, но оно никогда не наступит. Искать недостижимые состояния надо по порядку:

- обведи начальное состояние;
- обведи все состояния, куда ведёт стрелка из уже обведённых;
- повторяй, пока обводить нечего;
- необведённые – недостижимые.

Проверять «есть ли у состояния входящая стрелка» мало. Стрелка может приходить из другого недостижимого состояния: в подвал ведёт люк из погреба, а в погреб не ведёт ничего.

Мёртвая стрелка. Стрелка есть, но её условие никогда не бывает верным. Например, лифт в девятиэтажке и стрелка «приехал [этаж > 9] → Крыша». Десятого этажа в доме нет, и лифт по этой стрелке не пройдёт никогда. Всё, что лежит за ней, тоже может стать недостижимым.

Путаница двух стрелок. Из одного состояния выходят две стрелки с одним и тем же событием, и условия у них могут оказаться верными одновременно. Какая сработает? Машина «не знает», а программа поведёт себя непредсказуемо. Условия у стрелок с одним событием не должны пересекаться: $[t < 18]$ и $[t > 24]$ – хорошо, $[t < 20]$ и $[t > 17]$ – плохо, при 18 и 19 градусах подходят обе.

Стрелка не туда. По событию «прошло 10 с» турникет должен закрыться, а стрелка ведёт в «Тревогу». Такие ошибки находят только сравнением с заданием: прочитай, что должно происходить, и проверь каждую стрелку полной фразой, как советовали в разделе 6.2.

Нет нужной стрелки. В задании сказано «сирену выключают кнопкой Б», а стрелки с кнопкой Б из «Тревоги» нет. Мишка будет выть, пока не сядет батарейка.

Запомни. Тупик – нет выхода. Недостижимое – нет пути из начального (проверяй обводкой, а не поиском входящей стрелки). Две стрелки с одним событием из одного состояния – условия не должны пересекаться.

Совет. Проверая чужую схему, иди по заданию фраза за фразой и ставь галочку у стрелки, которая эту фразу выполняет. Стрелки без галочки – лишние или неправильные, фразы без стрелки – забытые переходы.

6.6. Как это выглядит в Кибериаде IDE

НА ПАЛЬЦАХ

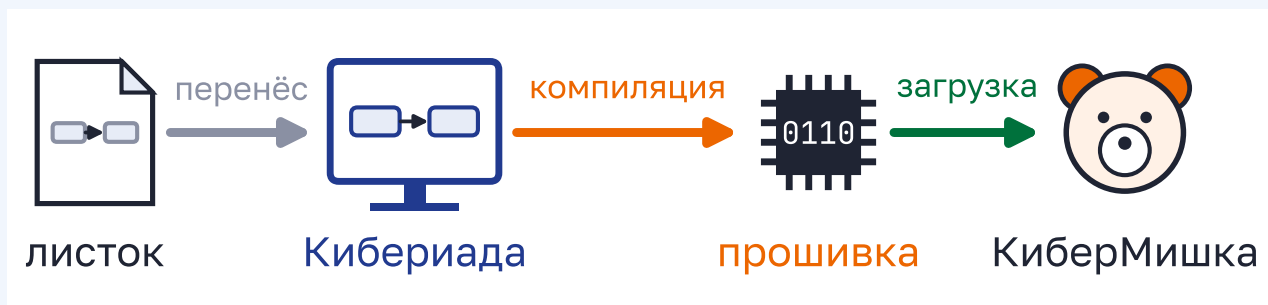


Рисунок 6.9 – Путь схемы: листок, Кибериада, компиляция, прошивка в КиберМишке

Схему сначала рисуешь на листке, потом ту же самую – в Кибериаде. Компиляция переводит её в команды для маленького компьютера внутри Мишки – получается прошивка. Её загружают в Мишку, и он живёт по твоей схеме.

Всё, что было выше, – это язык схем. В сфере «Кибертех» на нём программируют в **Кибериаде IDE**. IDE – это программа, в которой пишут и запускают другие программы (от английского «интегрированная среда разработки»). Кибериада графическая: в ней программу собирают из состояний и переходов, как на наших рисунках.

В программе кружка ПРИМС начинается с двух модулей:

- **модуль 5** – знакомство с Кибериадой IDE и понятия ПРИМС: диаграмма состояний, состояние, действие, переход, событие, иерархия (вложенные состояния). Там же – как построить машину состояний, скомпилировать её и загрузить;
- **модуль 6** – как обрабатываются события, условия и переходы.

Компилировать – значит превратить нарисованную схему в программу, которую поймёт микроконтроллер (маленький компьютер внутри КиберМишки). Готовую программу загружают в устройство, и она становится его **прошивкой**. На вебинаре разработчики сферы говорят о ПРИМС просто: это программирование устройств и знакомство с тем, что такое прошивка и как её загружают.

Настоящий КиберМишка для подготовки не нужен. В курсе сферы задачи решают в виртуальных симуляторах, а если оборудование есть – ещё и на КиберМишке.

Где искать кнопки? Не в этой книге: Кибериада обновляется, и окна от версии к версии меняются. Открой справку клавишей **F1** – справа появится панель, начни с раздела «Быстрый старт – КиберМишка – Начало» и листай его кнопками внизу панели. Как установить программу, написано в справочнике в конце книги.

Схема в Кибериаде та же, что на бумаге: состояния, стрелки, события, условия, действия, вложенность. Поэтому порядок работы советуем такой:

- сначала нарисуй машину на листке и проговори каждую стрелку полной фразой;
- потом прогони на листке трассировку для двух-трёх цепочек событий, в том числе для «неудобных» (событие, на которое нет стрелки; условие на границе);
- только после этого собирай схему в Кибериаде и запускай в симуляторе;

- если симулятор ведёт себя не так, как листок, ищи расхождение по таблице трассировки – строку, где разошлись.

Совет. Поставь Кибериаду заранее, не в день отбора: Windows может спросить разрешение на запуск, а для драйверов нужны права администратора. Тут пригодится взрослый.

6.7. Разбираем задачи

Задача 6.1 Лампа с тремя режимами

разминка · трассировка, событие без стрелки

У лампы одна кнопка и три режима: 1 – «Выкл», 2 – «Слабо», 3 – «Ярко». Короткое нажатие (К) переключает режимы по кругу: «Выкл» → «Слабо» → «Ярко» → «Выкл». Долгое нажатие (Д) выключает включённую лампу. Лампа выключена. Кнопку нажали так: К, К, Д, Д, К, К, К, К. а) В каком режиме лампа? Ответ – номер режима. б) Сколько нажатий ничего не изменили?

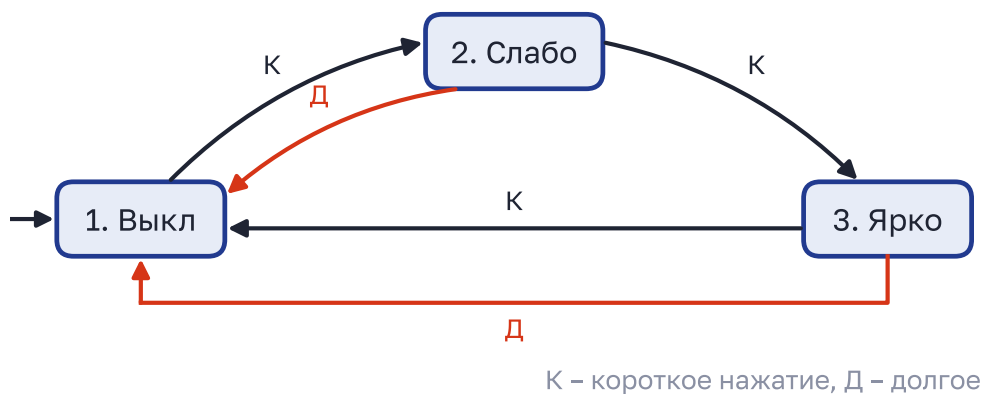


Рисунок 6.10 – Диаграмма лампы. Из «Выкл» стрелки «Д» нет

Дано: диаграмма лампы; начальное состояние «Выкл»; нажатия К, К, Д, Д, К, К, К, К.

Найти: состояние после восьмого нажатия; число нажатий без изменений.

Решение

Шаг 1. Выписываем стрелки. Из «Выкл»: К → «Слабо». Из «Слабо»: К → «Ярко», Д → «Выкл». Из «Ярко»: К → «Выкл», Д → «Выкл». Из «Выкл» стрелки Д нет.

Шаг 2. Трассировка.

Таблица 6.2

Шаг	Было	Нажатие	Стрелка	Стало
1	Выкл	К	есть	Слабо
2	Слабо	К	есть	Ярко
3	Ярко	Д	есть	Выкл
4	Выкл	Д	нет	Выкл

Шаг	Было	Нажатие	Стрелка	Стало
5	Выкл	К	есть	Слабо
6	Слабо	К	есть	Ярко
7	Ярко	К	есть	Выкл
8	Выкл	К	есть	Слабо

Шаг 3. Ответ на а). После восьмого нажатия лампа в «Слабо» – режим 2.

Шаг 4. Ответ на б). Без изменений прошло одно нажатие – четвёртое: долгое нажатие выключенной лампы.

Проверим концовку по-другому. После четвёртого нажатия лампа выключена, дальше четыре коротких подряд. Круг из коротких нажатий – три шага, $4 = 3 + 1$: целый круг возвращает в «Выкл», и ещё одно нажатие – «Слабо».

⚠ Типичная ошибка. На четвёртом нажатии решить, что долгое нажатие работает как выключатель туда и обратно и включает лампу в «Слабо». Стрелки Д из «Выкл» нет – ничего не происходит. С такой ошибкой дальше выходит «Ярко» → «Выкл» → «Слабо» → «Ярко», ответ 3, а пропавших нажатий – ни одного.

Ответ: а) 2 («Слабо»); б) 1.

Своя задача по программе сферы, модуль 5

Реши сам

6.1. У ёлочной гирлянды одна кнопка. Короткое нажатие (К) переключает режимы: «Выкл» → «Горит» → «Бегущий огонь» → «Мигает» → «Горит» → ... (обратно в «Выкл» короткое нажатие не ведёт). Долгое нажатие (Д) из любого включённого режима выключает гирлянду, в «Выкл» оно ничего не делает. Гирлянда выключена, нажали: К, К, К, К, Д, Д, К, К. В каком режиме гирлянда?

6.2. Лампу из задачи 6.1 включили из «Выкл» и нажали короткое нажатие 100 раз подряд. В каком режиме лампа? Сколько раз за это время она загоралась ярко?

Задача 6.2 Светофор с кнопкой у школы

таймеры, условие на стрелке, трассировка по времени

У пешеходного перехода возле школы светофор с кнопкой. Его машина состояний:

- «А. Машинам зелёный» – начальное. По событию «нажата кнопка» переходит в «Б», но только если в «А» светофор находится уже не меньше 20 с: [в «А» не меньше 20 с]. Иначе нажатие пропадает.
- «Б. Машинам жёлтый» – через 3 с переходит в «В».
- «В. Пешеходам зелёный» – через 15 с переходит в «Г».

- «Г. Пешеходам мигает» – через 4 с переходит в «А».

В «Б», «В» и «Г» стрелок по нажатию кнопки нет. Светофор включили в момент 0 с. Кнопку нажимали в моменты 8, 25, 30, 41, 52, 70 и 90 с. Сколько раз за первые 100 с пешеходам загорался зелёный и в какой момент – впервые? Ответ – два числа через пробел.

Дано: состояния А (условие 20 с), Б (3 с), В (15 с), Г (4 с); нажатия в 8, 25, 30, 41, 52, 70, 90 с.

Найти: сколько раз наступало «В» до 100 с; момент первого входа в «В».

Решение

Шаг 1. Что считать. Для каждого нажатия нужно знать, в каком состоянии светофор, а если в «А» – сколько секунд он там. Добавим в таблицу столбец «в состоянии с» – момент, когда светофор вошёл в текущее состояние.

Шаг 2. Нажатие в 8 с. Светофор в «А» с момента 0. В «А» он $8 - 0 = 8$ с, а нужно не меньше 20. Условие неверно, нажатие пропало.

Шаг 3. Нажатие в 25 с. Всё ещё «А» с момента 0: $25 - 0 = 25$ с, это не меньше 20. Переход в «Б» в 25 с. Дальше таймеры: «В» начнётся в $25 + 3 = 28$ с, «Г» – в $28 + 15 = 43$ с, «А» – в $43 + 4 = 47$ с.

Шаг 4. Нажатия в 30 и 41 с. С 28 по 43 с светофор в «В». Из «В» стрелки по кнопке нет – оба нажатия пропали.

Шаг 5. Нажатие в 52 с. В «А» светофор вернулся в 47 с. Прошло $52 - 47 = 5$ с – меньше 20, пропало.

Шаг 6. Нажатие в 70 с. В «А» с 47 с: $70 - 47 = 23$ с – условие верно. «Б» в 70 с, «В» в $70 + 3 = 73$ с, «Г» в $73 + 15 = 88$ с, «А» в $88 + 4 = 92$ с.

Шаг 7. Нажатие в 90 с. Светофор в «Г» (88–92 с) – пропало.

Таблица 6.3

Нажатие, с	Состояние	В нём с, с	Прошло, с	Итог
8	А	0	8	пропало
25	А	0	25	переход
30	В	28	–	нет стрелки
41	В	28	–	нет стрелки
52	А	47	5	пропало
70	А	47	23	переход
90	Г	88	–	нет стрелки

Шаг 8. Считаем. Пешеходам зелёный загорался в 28 с и в 73 с – два раза. К 100 с светофор снова в «А» (с 92 с).

Из семи нажатий сработали два. Ничего странного: у настоящих светофоров с кнопкой нетерпеливые пешеходы жмут много раз, а светофор всё равно выдерживает машинам минимальное время.

❗ Типичная ошибка. Считать 20 с от момента включения, а не от последнего входа в «А». Тогда нажатие в 52 с «проходит» ($52 > 20$), зелёный загорается в 55 с, и весь хвост трассировки уезжает. Другая ошибка – «запомнить» нажатие в 30 с и выполнить его, когда светофор вернётся в «А». Машина состояний ничего не запоминает, если это не нарисовано на схеме: нет стрелки – нет события.

Ответ: 2 28 (два раза, впервые в 28 с).

Своя задача по программе сферы, модуль 6

Реши сам

6.3. Другой светофор устроен так же, но с другими временами: переход из «А» по кнопке – только если в «А» не меньше 15 с; «Б» длится 2 с, «В» – 12 с, «Г» – 3 с. Кнопку нажимали в 6, 18, 25, 33, 40 и 55 с. Сколько нажатий пропали впустую и в какой момент машинам последний раз загорелся зелёный («А») за первые 90 с?

6.4.* Светофор из задачи 6.2. Пешеход нажимает кнопку в самые удачные моменты. Какое наибольшее число раз пешеходам может загореться зелёный за первые 120 с после включения? Здесь цепочка событий не дана – её придумываешь ты. Сначала реши, когда выгоднее всего нажать: чуть позже или в первую же секунду, когда условие станет верным.

Задача 6.3 Турникет на входе в школу

ошибка в диаграмме, сравнение с заданием

Турникет должен работать так.

- Три состояния: «Закрит», «Открыт», «Тревога». Начальное – «Закрит».
- К закрытому турникету приложили карту. Если карта своя, турникет открывается и зажигает зелёную стрелку. Если чужая, пищит и остаётся закрытым.
- Человек прошёл через открытый турникет – турникет закрывается.
- Открытый турникет простоял 10 с, и никто не прошёл, – закрывается.
- Закрытый турникет пытаются продавить – «Тревога» и сирена.
- Из «Тревоги» выводит только ключ охранника, после него турникет «Закрит».

Программист записал такие переходы:

1. Закрит – карта [своя] / зелёная стрелка → Открыт
2. Закрит – карта [чужая] / писк → Открыт
3. Открыт – прошёл → Закрит
4. Открыт – прошло 10 с → Тревога
5. Закрит – толкнули / сирена → Тревога
6. Тревога – карта [своя] → Закрит
7. Тревога – ключ охранника → Закрит

Какие переходы записаны с ошибкой? Ответ – номера по возрастанию, без пробелов.

Дано: шесть требований; семь переходов.

Найти: номера неправильных переходов.

Решение

Шаг 1. Каждый переход – полной фразой, и сверяем с заданием.

- Переход 1: «Если закрыт и приложили свою карту – зажечь стрелку и открыться». В задании так и есть.
- Переход 2: «Если закрыт и карта чужая – пискнуть и **открыться**». В задании: пищит и **остаётся закрытым**. Ошибка: стрелка должна вести в «Закрыт» (или её можно вовсе не рисовать, если писк не нужен). Турникет пускает всех подряд – ошибка серьёзная.
- Переход 3: «Если открыт и человек прошёл – закрыться». Совпадает с заданием.
- Переход 4: «Если открыт и прошло 10 с – **тревога**». В задании: закрывается. Ошибка: стрелка не туда. Ученик приложил карту, задумался – и школа слышит сирену.
- Переход 5: «Если закрыт и толкнули – сирена и тревога». Тут всё как надо.
- Переход 6: «Если тревога и приложили свою карту – закрыться». В задании: из «Тревоги» выводит **только** ключ охранника. Ошибка: лишняя стрелка. С ней любой ученик сам выключит тревогу своей картой.
- Переход 7: «Если тревога и ключ охранника – закрыться». Ровно это и требуется.

Шаг 2. Проверяем обратное: у каждой фразы задания есть своя стрелка? Своя карта – 1. Чужая карта – 2 (неправильная). Прошёл – 3. 10 с – 4 (неправильная). Толкнули – 5. Ключ охранника – 7. Забытых переходов нет.

Шаг 3. Собираем ответ. Ошибки в переходах 2, 4 и 6.

 **Типичная ошибка.** Пропустить переход 6: «своя карта же – значит, можно». Слово «только» в задании запрещает любые другие стрелки из «Тревоги». Ещё одна – читать лишь, куда ведёт стрелка, и не смотреть, откуда она выходит: переходы 1 и 6 отличаются только началом.

Ответ: 246.

Своя задача по программе сферы, модуль 6

Реши сам

6.5. Гаражные ворота: «Закрыты» (начальное), «Открываются», «Открыты», «Закрываются». Задание: кнопка пульта в «Закрыты» – начать открывать; когда ворота доехали до верха – «Открыты»; кнопка пульта в «Открыты» – начать закрывать; доехали до низа – «Закрыты»; если, пока ворота закрываются, датчик увидел препятствие (машину, кошку) – снова открываются. Переходы: 1) Закрыты – пульт → Открываются; 2) Открываются – доехали до верха → Закрываются; 3) Открыты – пульт → Закрываются; 4) Закрываются – доехали до низа → Закрыты; 5) Закрываются – препятствие → Закрыты. Какие переходы неправильные? Ответ – номера по возрастанию, без пробелов.

6.6. Автоматика теплицы на даче в состоянии «Жду» раз в 10 минут измеряет температуру воздуха t (событие «замер»). По заданию при t ниже 15°C она должна перейти в «Греет», при t выше 28°C – в «Проветривает» (открыть форточку), при остальных – ждать дальше. Программист нарисовал из «Жду» две стрелки: «замер [$t < 22$] → Греет» и «замер [$t > 19$] → Проветривает». При каких целых температурах автоматика «не знает», по какой стрелке идти? Как исправить условия?

Задача 6.4 КиберМишка сторожит штаб

дополнить переход, условие, трассировка

КиберМишка охраняет дверь в штаб. Его схема такая.

- «Жду» – начальное, экран тёмный. Кнопка А – переход в «Слежу».
- «Слежу» – на экране глаз. Звук громче 70 – переход в «Тревога», экран красный, сирена. Кнопка А – обратно в «Жду».
- «Тревога» – стрелок из неё пока нет.

По заданию в «Тревоге» должно быть так: кнопка Б выключает сирену, и Мишка снова следит; если кнопку никто не нажал, через 30 с Мишка сам выключает сирену и снова следит. Тихие звуки (громкость 70 и меньше) ничего не меняют ни в каком состоянии.

Какие из стрелок надо добавить?

1. Тревога – кнопка А / выключить сирену → Жду
2. Тревога – кнопка Б / выключить сирену → Слежу
3. Слежу – звук [громкость ≤ 70] / сирена → Тревога
4. Тревога – прошло 30 с → Жду
5. Тревога – прошло 30 с / выключить сирену → Слежу
6. Жду – звук [громкость > 70] / сирена → Тревога

Потом проследи за исправленным Мишкой. События: кнопка А, звук 65, звук 80, прошло 30 с, звук 90, кнопка Б, кнопка А. В каком состоянии Мишка в конце? Ответ: номера нужных стрелок по возрастанию без пробелов и название состояния.

Дано: схема без выходов из «Тревоги»; требования; шесть стрелок-кандидатов; цепочка из семи событий.

Найти: какие стрелки добавить; конечное состояние.

Решение

Шаг 1. Что требуется из «Тревоги». Две фразы задания: «кнопка Б – выключить сирену и следить» и «через 30 с – выключить сирену и следить». Значит, нужны две стрелки из «Тревоги» в «Слежу».

Шаг 2. Проверяем кандидатов.

- Стрелка 1 ведёт в «Жду» по кнопке А. В задании выключает кнопка Б и Мишка продолжает следить. Не подходит.
- Стрелка 2 – кнопка Б, выключить сирену, в «Слежу». Подходит.

- Стрелка 3 поднимает тревогу от тихого звука. Задание прямо запрещает. Не подходит.
- Стрелка 4 – событие правильное, но ведёт в «Жду» и сирену не выключает. Не подходит: Мишка перестанет следить, а сирена продолжит выть.
- Стрелка 5 – через 30 с, выключить сирену, в «Слежу». Подходит.
- Стрелка 6 поднимает тревогу из «Жду», когда охрана выключена. Не подходит.

Добавить надо 2 и 5.

Шаг 3. Трассировка исправленного Мишки.

Таблица 6.4

Шаг	Было	Событие	Стрелка	Стало
1	Жду	кнопка А	есть	Слежу
2	Слежу	звук 65	условие $65 > 70$ неверно	Слежу
3	Слежу	звук 80	$80 > 70$, есть	Тревога
4	Тревога	прошло 30 с	стрелка 5	Слежу
5	Слежу	звук 90	$90 > 70$, есть	Тревога
6	Тревога	кнопка Б	стрелка 2	Слежу
7	Слежу	кнопка А	есть	Жду

Мишка в «Жду»: последним нажатием охрану выключили.

 **Типичная ошибка.** Взять стрелку 4, потому что «событие то же». Сверяй все три части подписи и конец стрелки: событие, условие, действие и куда она ведёт. На шаге 2 трассировки ошибаются иначе: звук есть – значит, тревога. Но у стрелки условие, и 65 не больше 70.

Ответ: 25; Мишка в «Жду».

Своя задача по программе сферы, модуль 6

Реши сам

6.7. КиберМишка-будильник: «Спит» (начальное), «Звенит», «Отложен». Нарисованы стрелки: Спит – наступило 7:00 → Звенит; Звенит – кнопка А → Отложен; Звенит – кнопка Б → Спит. По заданию отложенный будильник через 5 минут звенит снова. Какую стрелку надо добавить: 1) Звенит – прошло 5 мин → Отложен; 2) Отложен – прошло 5 мин → Звенит; 3) Отложен – кнопка А → Звенит; 4) Спит – прошло 5 мин → Звенит? Никита каждый раз, как будильник зазвенит, жмёт А, а на третий звонок жмёт Б. Во сколько будильник зазвенел последний раз?

6.8. Мишке из задачи 6.4 (уже исправленному) добавили состояние «Выключен». Долгое нажатие кнопки А в любом из трёх состояний выключает Мишку. Короткое нажатие А в «Выключен» включает его, и Мишка оказывается в «Жду». Старые стрелки по кнопке А («Жду» – «Слежу» и обратно) остаются как были и теперь срабатывают от короткого нажатия. Сколько новых стрелок придётся нарисовать без вложенных состояний? А если положить «Жду», «Слежу» и «Тревога» в рамку «Включён»?

Задача 6.5 Робот-пылесос с ошибками

тупики, недостижимые состояния, обводка

Школьники нарисовали машину состояний робота-пылесоса. По замыслу робот работает бесконечно: ни одно состояние не должно быть последним, и в каждое можно попасть. Найди недостижимые состояния и тупики. Ответ – номера недостижимых по возрастанию без пробелов, затем через пробел номера тупиков.

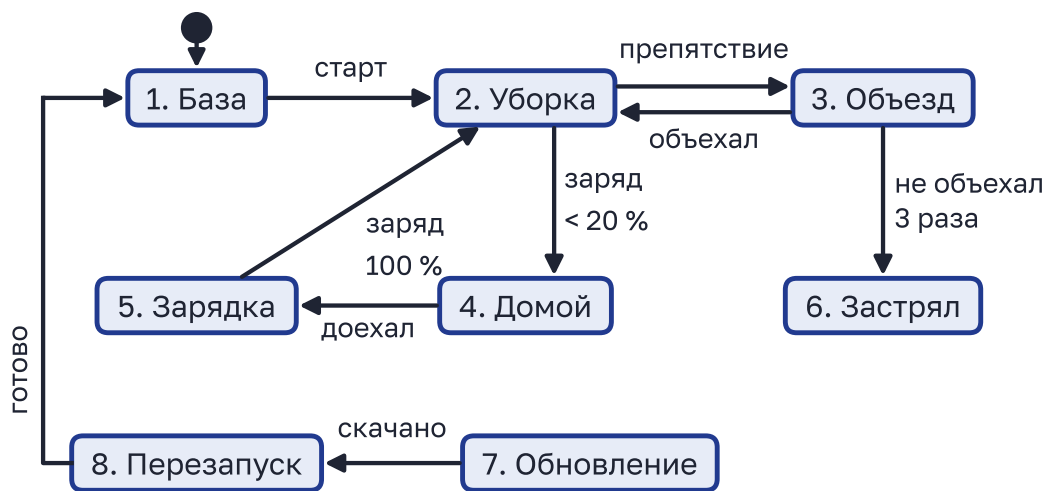


Рисунок 6.11 – Схема робота-пылесоса. Начальное состояние – «База»

Дано: 8 состояний и 9 стрелок на схеме; начальное – 1.

Найти: недостижимые состояния; тупики.

Решение

Шаг 1. Выписываем стрелки списком. 1 → 2 (старт); 2 → 3 (препятствие); 3 → 2 (объехал); 2 → 4 (заряд < 20 %); 4 → 5 (доехал); 5 → 2 (заряд 100 %); 3 → 6 (не объехал 3 раза); 7 → 8 (скачано); 8 → 1 (готово).

Шаг 2. Обводка. Начинаем с 1. Из 1 – в 2, обвели 2. Из 2 – в 3 и 4, обвели. Из 3 – в 2 (уже обведено) и в 6, обвели 6. Из 4 – в 5, обвели. Из 5 – в 2, уже есть. Из 6 стрелок нет. Новых нет, стоп. Обведены: 1, 2, 3, 4, 5, 6.

Шаг 3. Недостижимые. Не обведены 7 и 8. Хитрее всего с 8: в него входит стрелка из 7. Но в 7 не входит ничего, значит, и 8 не наступит никогда. Кнопку «обновить прошивку» школьники нарисовать забыли.

Шаг 4. Тупики. Смотрим, из каких состояний не выходит ни одной стрелки. Из 1 выходит, из 2 – три, из 3 – две, из 4, 5, 7, 8 – по одной. Из 6 – ни одной: робот застрял и будет стоять, пока не сядет батарея. Нужна стрелка вроде «подняли и поставили на пол → Уборка».

Получилось два недостижимых и один тупик. Для схемы, которую рисовали в спешке, картина правдоподобная: забыли вход в новую ветку и выход из аварийного состояния.

 **Типичная ошибка.** Искать недостижимые по входящим стрелкам. Тогда 8 кажется достижимым – в него же ведёт стрелка. Проверяй обводкой от начального. Вторая ошибка – записать 1 в тупики или в недостижимые из-за того, что в «Базу» идёт стрелка только из недостижимого 8. Но 1 – начальное: машина в нём с самого начала.

Ответ: 78 6.

Своя задача по программе сферы, модуль 5

Реши сам

6.9. Персонаж игры: 1 – «Стоит» (начальное), 2 – «Идёт», 3 – «Бежит», 4 – «Прыгает», 5 – «Плывёт», 6 – «Тонет», 7 – «Летит». Переходы: 1 → 2, 2 → 1, 2 → 3, 3 → 2, 3 → 4, 2 → 5, 5 → 6, 7 → 4. По замыслу ни одно состояние не последнее. Найди недостижимые состояния и тупики. Ответ – в том же виде, что в задаче 6.5.

6.10. Электрический чайник: 1 – «Выкл» (начальное), 2 – «Греет», 3 – «Готово». Переходы: Выкл – кнопка → Греет; Греет – кнопка → Выкл; Греет – замер [температура > 100 °C] / щелчок → Готово; Готово – прошло 30 мин → Выкл. Вода в открытом чайнике не нагревается выше 100 °C. Какое состояние никогда не наступит? Где окажется чайник, если нажать кнопку и уйти на час?

Задача 6.6 * Музыкальный Мишка

вложенные состояния, событие передаётся родителю, трассировка

КиберМишка играет музыку и получает по инфракрасному лучу команды 0, 1 и 2. Внутри рамки «Игра» три состояния: «Тихо» (начальное), «Громко», «Эхо». Снаружи – «Пауза». Стрелки:

- внутри: Тихо – 0 → Громко; Громко – 0 → Эхо; Эхо – 0 → Тихо; Громко – 1 → Тихо; Эхо – 2 → Громко;
- от рамки «Игра»: 1 → Пауза; 2 → Тихо;
- из «Паузы»: 1 → в рамку «Игра» (то есть в её начальное состояние «Тихо»). На 0 и 2 «Пауза» не отвечает.

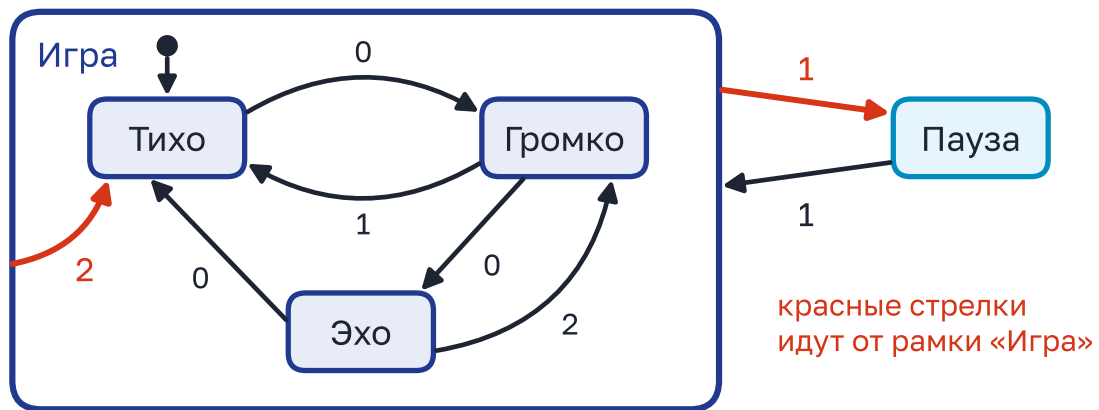


Рисунок 6.12 – Музыкальный Мишка. Красные стрелки начинаются на рамке «Игра»

Мишку включили, он в «Тихо». Пришли команды: 0, 1, 0, 0, 2, 2, 0, 0, 1, 0, 1, 0, 0. В каком состоянии Мишка в конце и сколько раз он попадал в «Паузу»?

Дано: схема с рамкой «Игра»; 13 команд.

Найти: конечное состояние; число входов в «Паузу».

Решение

Шаг 1. Правило для каждой команды. Сначала ищем стрелку у текущего вложенного состояния. Нет её – смотрим на рамку. Нет и у рамки – команда пропала. В «Паузе» рамки нет, отвечает только сама «Пауза».

Шаг 2. Кто что умеет. «Тихо» отвечает только на 0, на 1 и 2 за него отвечает рамка. «Громко» отвечает на 0 и на 1 сам, на 2 – рамка. «Эхо» отвечает на 0 и 2 сам, на 1 – рамка. Здесь ловушка: у «Громко» своя стрелка по 1, поэтому из «Громко» команда 1 ведёт в «Тихо», а в «Паузу» не ведёт.

Шаг 3. Трассировка. В столбце «кто» – чья стрелка сработала.

Таблица 6.5

Шаг	Было	Команда	Кто	Стало
1	Тихо	0	Тихо	Громко
2	Громко	1	Громко	Тихо
3	Тихо	0	Тихо	Громко
4	Громко	0	Громко	Эхо
5	Эхо	2	Эхо	Громко
6	Громко	2	рамка	Тихо
7	Тихо	0	Тихо	Громко
8	Громко	0	Громко	Эхо
9	Эхо	1	рамка	Пауза
10	Пауза	0	никто	Пауза
11	Пауза	1	Пауза	Тихо

Шаг	Было	Команда	Кто	Стало
12	Тихо	0	Тихо	Громко
13	Громко	0	Громко	Эхо

Шаг 4. Итог трассировки. Мишка в «Эхо». В «Паузу» он попал один раз – на шаге 9. На шаге 11 он вернулся в рамку и по правилу входа оказался в «Тихо», хотя уходил из «Эхо».

Обе команды 2 на шагах 5 и 6 дали разный результат. Из «Эхо» двойка ведёт в «Громко», из «Громко» – через рамку в «Тихо». Одна и та же команда, а что она значит – зависит от того, где машина сейчас.

 **Типичная ошибка.** Сначала смотреть на рамку, а потом на вложенное состояние. Тогда уже на шаге 2 команда 1 уводит в «Паузу», и в конце выходит «Пауза». Порядок обратный: первым отвечает самое внутреннее состояние. Вторая ошибка – на шаге 11 вернуться туда, откуда ушли («Эхо»). Вход в рамку всегда ведёт в её начальное состояние.

Ответ: «Эхо»; в «Паузе» – 1 раз.

По мотивам задачи второго отборочного этапа НТО 2025/26, профиль «Интеллектуальные энергетические системы», «Киберфизика по Маркову»

Реши сам

6.11. Тот же музыкальный Мишка в «Тихо». Команды: 2, 0, 1, 0, 0, 1, 2, 1, 0, 2, 0. В каком состоянии Мишка в конце и сколько раз он попадал в «Паузу»?

6.12. У робота-тележки рамка «Работа» с четырьмя вложенными состояниями: «Жду», «Еду», «Поворачиваю», «Сигналю». От рамки выходят две стрелки: «стоп» → «Стою» и «разряд» → «Сплю». У «Сигналю» есть и своя стрелка «стоп» → «Жду», и она остаётся. Робот должен вести себя так же, но без рамки. Сколько стрелок придётся нарисовать вместо двух стрелок от рамки?

Задача 6.7* Кодовый замок 3-1-3

кодовый замок, распознавание последовательности, исправление схемы

Петя собрал из КиберМишки кодовый замок. Замок открывается, если три последние нажатые цифры – 3, 1, 3. Петя нарисовал схему: «Пусто» (начальное) – 3 → «3»; «3» – 1 → «3-1»; «3-1» – 3 → «Открыт». Любая другая цифра из «Пусто», «3» и «3-1» возвращает в «Пусто».

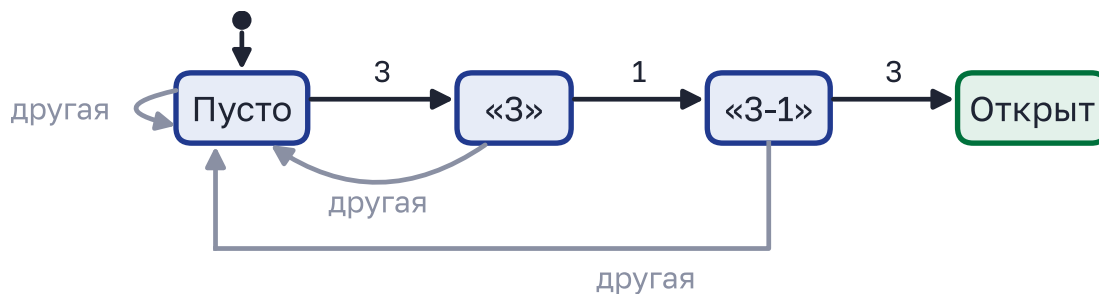


Рисунок 6.13 – Схема Пети. «Другая» – любая цифра, кроме подписанной на соседней стрелке

- а) Нажали цифры 2, 3, 3, 1, 3, 1, 3. На каком по счёту нажатии замок Пети откроется?
- б) Какую одну стрелку надо поменять, чтобы замок открывался всегда, как только последние три цифры – 3, 1, 3? 1) из «3» по цифре 1 – в «Пусто»; 2) из «3» по цифре 3 – в «3»; 3) из «3-1» по цифре 3 – в «3»; 4) из «Пусто» по цифре 3 – в «3-1».
- в) На каком нажатии откроется исправленный замок при тех же цифрах?
- Ответ – три числа через пробел.

Дано: схема Пети; цифры 2, 3, 3, 1, 3, 1, 3; четыре варианта исправления.

Найти: номер нажатия для схемы Пети; номер варианта; номер нажатия для исправленной схемы.

Решение

Шаг 1. Трассировка схемы Пети.

Таблица 6.6

Нажатие	Было	Цифра	Стало
1	Пусто	2	Пусто
2	Пусто	3	«3»
3	«3»	3	Пусто (другая)
4	Пусто	1	Пусто
5	Пусто	3	«3»
6	«3»	1	«3-1»
7	«3-1»	3	Открыт

Замок Пети откроется на 7-м нажатии. А ведь после 5-го нажатия последние три цифры уже были 3, 1, 3! Замок их пропустил.

Шаг 2. Где ломается. На 3-м нажатии замок был в «3» и получил ещё одну тройку. По схеме Пети это «другая цифра», и замок сбросился в «Пусто». Но последняя нажатая цифра – 3, а код с тройки и начинается. Правильно остаться в «3»: «код, может быть, начинается с этой тройки».

Шаг 3. Проверяем варианты. Вариант 1 ломает основной путь 3 → 1. Вариант 3 неверен: из «3-1» тройка открывает замок, и эту стрелку трогать нельзя. Вариант 4 перепрыгивает через единицу. Подходит вариант 2: из «3» по цифре 3 – в «3».

Хватит ли одной этой стрелки? Проверим остальные «другие» цифры. Из «3-1» пришла не тройка: последние цифры «3, 1, х», и ни «1, х», ни «х» не начинают код – сброс в «Пусто» верен. Из «3» пришла не 1 и не 3 – тоже сброс. Из «Пусто» не тройка – остаётся в «Пусто». Других исправлений не нужно. Этот вывод мы ещё проверили программой: перебрали все цепочки до восьми цифр из 0, 1, 2, 3, и исправленная схема ни разу не разошлась с правилом «три последние – 3, 1, 3».

Шаг 4. Исправленный замок. 1: 2 – «Пусто». 2: 3 – «3». 3: 3 – остаётся в «3». 4: 1 – «3-1». 5: 3 – «Открыт». Откроется на 5-м нажатии, как только последние три цифры стали 3, 1, 3.

 **Типичная ошибка.** Думать, что «любая неверная цифра – начинай сначала» – это всегда правильно. Не всегда: неверная для текущего места цифра может оказаться началом нового кода. У кодов вроде 1-2-3, где цифры не повторяются, такой беды нет. А у 3-1-3 или 1-1-2 она есть – ищи её в первую очередь.

Ответ: 7 2 5.

Своя задача по программе сферы, модуль 6

Реши сам

6.13. Замок с кодом 2-5 нарисован так же, как у Пети: «Пусто» – 2 → «2»; «2» – 5 → «Открыт»; любая другая цифра – в «Пусто». Нажали 2, 2, 5, 7, 2, 5. На каком нажатии откроется такой замок? На каком откроется правильный, который открывается сразу, как только две последние цифры – 2, 5?

6.14.* Код замка – 1-1-2. Состояния: «Пусто», «1», «1-1», «Открыт». В какое состояние должна вести стрелка из «1-1» по цифре 1, чтобы замок работал правильно? На каком нажатии откроется правильный замок, если нажали 1, 2, 1, 1, 1, 2? А замок, у которого любая неверная цифра ведёт в «Пусто»?

Управление КиберМишкой



КиберМишка – главный герой сферы: учебный медвежонок-контроллер. Он слышит хлопки, ловит невидимый инфракрасный свет и отвечает цветом глаз, картинкой на животе и писком. В этой главе ты будешь им командовать, а заодно двумя исполнителями из симулятора Кибериады – Садовником и Строчником. Садовник в курсе сферы появляется раньше всех: он живёт в модулях 5 и 6, открытых до отбора, и за него в задатнике Кибериады дают самые дорогие задачи. КиберМишка и Строчник – темы более поздних модулей (в открытых модулях курса Строчника нет, он встречается в задатнике Кибериады). Вопросы «где окажется исполнитель», «что он сделает, если придёт такой сигнал» и «сколько команд нужно» решаются карандашом в тетради, и навык этот пригодится уже на отборе. Ответ в таких задачах – число, клетка, набор цифр или название состояния.

7.1. Исполнитель и его команды

НА ПАЛЬЦАХ

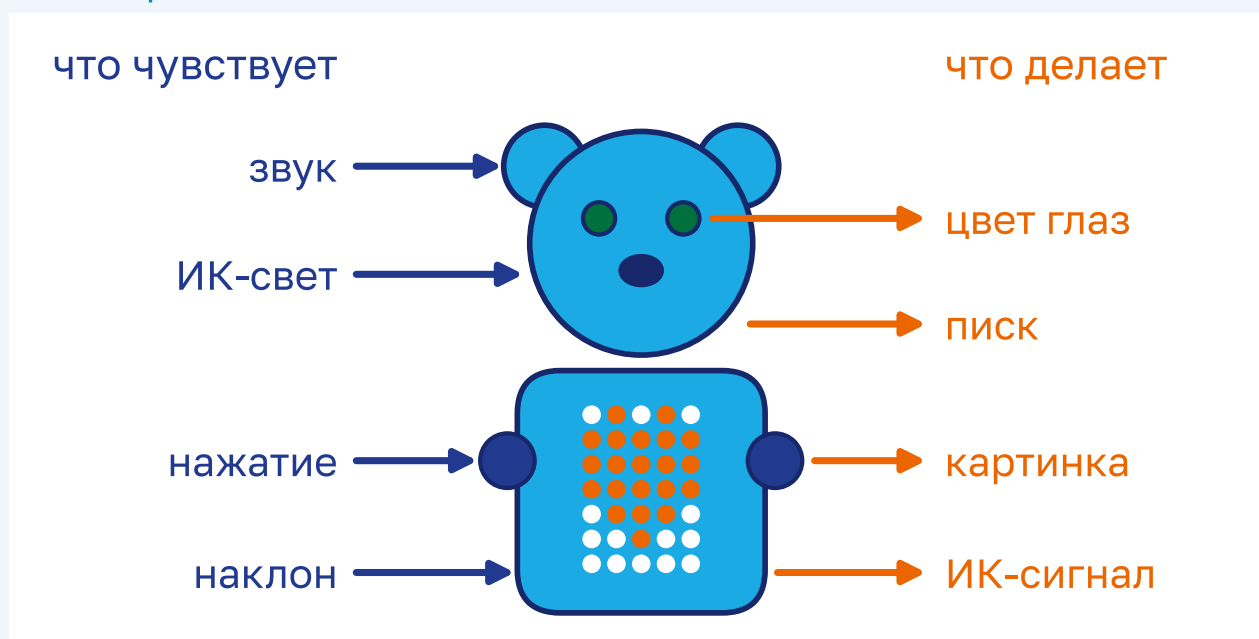


Рисунок 7.1 – КиберМишка: слева – что он чувствует, справа – что умеет делать

Всё, что приходит к Мишке снаружи, ловят датчики. Всё, что он делает сам, – действия. Программа стоит между ними и решает, на что как отвечать.

Исполнитель – тот, кто выполняет команды. Ты уже встречал исполнителей на уроках информатики: Черепашку, Робота, Чертёжника. У каждого исполнителя есть свой короткий список того, что он умеет. Этот список называют **системой команд исполнителя**. Команды из списка он выполнит в точности, а любую другую не поймёт, как бы вежливо ты ни просил. Собака, которая знает «сидеть» и «лежать», на «принеси тапки» только посмотрит на тебя с сочувствием.

КиберМишка – учебный контроллер в виде медвежонка. Сам он никуда не ездит. Он слушает, ловит инфракрасный свет, чувствует, как его повернули и на какую кнопку нажали, а в ответ меняет цвет глаз, рисует картинку на животе и пищит. Программу для него рисуют в Кибериаде IDE, а если настоящего Мишки нет, её запускают в симуляторе.

В Кибериаде Мишка собран из **компонентов** – деталей, каждая из которых отвечает за что-то одно. У компонента бывает три вида «ручек»:

- **события** – что компонент сообщает программе: «Клик» у кнопки, «Услышан громкий звук» у ушей;
- **действия** (команды) – что компонент делает по приказу программы: «Выбрать из палитры» у глаз;
- **значения** – числа, которые можно прочитать: «Яркость» у носа, «Значение» у счётчика.

Компоненты, которые ловят что-то снаружи, – датчики Мишки:

Таблица 7.1

Компонент	Что ловит	Событие
Уши-громкость	громкий звук	«Услышан громкий звук»
Уши-байты	байт, переданный звуком	«Получен байт»
Нос	инфракрасный свет	«Выше порогового значения», «Ниже порогового значения»
Нос-байты	байт, переданный ИК-лучом	«Получен байт»
Кнопка	нажатие лапы, пальца или бока	«Клик», «Зажата», «Отжата»
Акселерометр	как Мишку положили	«Положение изменилось»
Таймер	прошло заданное время	«Тайм-аут»

Нос у Мишки чувствует только **инфракрасный** свет – тот, которого не видит глаз, но у многих телефонов видит камера (направь на неё пульт от телевизора и нажми кнопку – на экране замигает огонёк). Яркость нос выдаёт числом от 0 до 4095. Громкость уши меряют по такой же шкале. Кнопок шесть: на правой и левой лапе, на правом и левом пальце, на правом и левом боку. Акселерометр отличает шесть положений: на ногах, на ушах, на левом боку, на правом боку, на спине, на животе.

Компоненты, которые что-то делают:

Таблица 7.2

Компонент	Главная команда	Что получается
Глаза	«Выбрать из палитры»	глаз загорается одним из 13 цветов
Матрица-картинка	«Установить картинку»	готовый рисунок на животе
Матрица светодиодов	«Яркость светодиода»	один огонёк нужной яркости
Пищалка	«Настроить частоту», «Запустить звук»	писк заданной высоты и длины
ИК-светодиод	«Отправить байт»	байт уходит невидимым лучом

Готовых картинок семь: сердце, улыбка, грусть, крест, нота, ромб, квадрат. Матрица на животе – это 35 светодиодов, 7 строк по 5 штук, и каждый можно зажечь отдельно с яркостью от 0 до 100. Частоту писка задают в герцах, до 24 000 Гц, а длительность – в миллисекундах.

Есть и помощники, которые ничего не чувствуют и ничего не показывают, зато помнят и считают. **Счётчик** хранит целое число: его можно увеличить, уменьшить, установить или сбросить в ноль. **Калькулятор** умеет прибавлять, вычитать, умножать, делить нацело и находить остаток от деления. **Генератор случайностей** бросает кубик.

Совет. Названия компонентов и команд здесь – как в Кибериаде IDE осенью 2026 года, у платформы «КиберМишка (a12)». В учебной платформе «(Junior) КиберМишка» набор короче: глаза, матрица, уши-байты, нос-байты, счётчик, калькулятор и «своё событие». Программа обновляется – если название не совпало, загляни в справку (F1).

Кроме Мишки, в Кибериаде есть исполнители, которые живут прямо в симуляторе. **Садовник** – робот на клетчатом поле. Он понимает такие команды:

- **«Вперёд»** – шаг на одну клетку туда, куда он смотрит;
- **«Назад»** – шаг на одну клетку в обратную сторону, **не поворачиваясь**;
- **«Повернуть налево»**, **«Повернуть направо»** – поворот на месте на четверть круга;
- **«Посадить»** – посадить в клетку розу, мяту или василёк;
- **«Осмотреть стены»** – датчик сообщает, есть ли стена спереди, справа, слева и сзади.

Ещё у Садовника есть компас: он знает свои координаты X и Y и сторону света, куда смотрит. По краям поля стоят перегородки. Садовник, который шагнул в перегородку, ломается, и программа останавливается с ошибкой. Тем же кончится попытка посадить цветок в занятую клетку.

Клетки поля нумеруют с нуля. Какой угол считать клеткой $(0; 0)$, смотри в условии задачи: в разных задачах бывает по-разному. В этой главе договоримся так: $(0; 0)$ – левый верхний угол, север – вверх, первое число x – номер столбца слева направо, второе y – номер строки **сверху вниз**. Значит, шаг на восток прибавляет 1 к x , на запад – отнимает. Шаг на юг прибавляет 1 к y , а шаг на север **отнимает 1** от y . Последнее непривычно, в тетради по математике ось y смотрит вверх. Здесь она смотрит вниз, как строки в таблице.

Третий исполнитель – **Строчник**. Он читает строку по одному символу слева направо и умеет подавать импульс на один из трёх выходов: А, Б или В. Например, в учебной задаче Кибериады он должен подать импульс А на каждую букву «А» в строке.

Запомни. Исполнитель выполняет только команды из своего списка. У КиберМишки датчики присылают **события** и **значения**, а глаза, матрица, пищалка и ИК-светодиод выполняют **действия**. Садовник: «Вперёд» и «Назад» – на клетку без поворота, повороты – на месте; на север y уменьшается.

7.2. Алгоритм управления: схема из состояний

НА ПАЛЬЦАХ

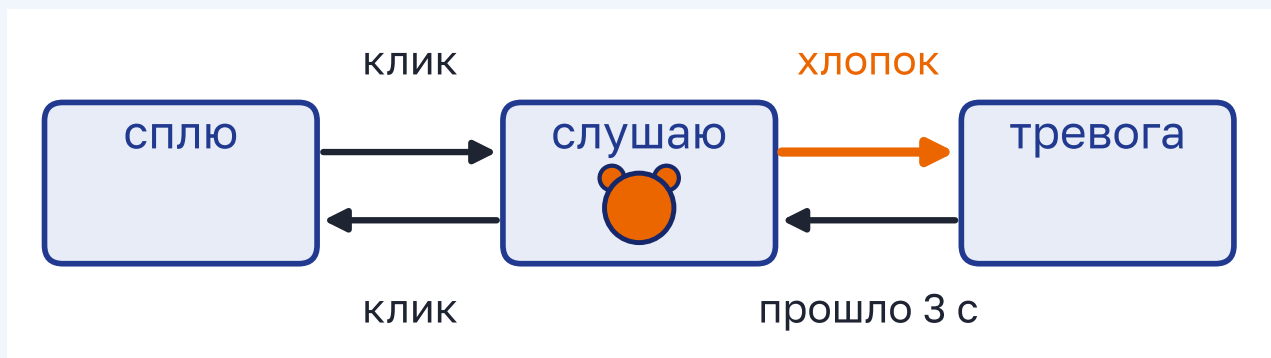


Рисунок 7.2 – Мишка – фишка в одной из комнат. Каждая дверь открывается только своим событием

Сейчас Мишка в комнате «слушаю». Кликнули по лапе – он уходит спать, хлопнули в ладоши – в «тревогу». Тайм-аут в этой комнате ничего не откроет: двери с такой надписью отсюда нет.

Программу для Мишки не пишут строчками, её рисуют схемой из **состояний** и **переходов** – с ними ты познакомился в главе 6, напомним коротко. **Состояние** – режим, в котором сейчас устройство: «сплю», «слушаю», «показываю». **Событие** – то, что случилось: кнопку кликнули, пришёл байт, прошло 3 секунды. **Переход** – стрелка из одного состояния в другое, на ней написано, какое событие её «открывает». Подпись на стрелке читается так: **событие [условие] / действие**. Условие в квадратных скобках и действие после косой черты бывают не всегда.

У состояния могут быть свои действия: **на входе** (выполняются один раз, когда машина вошла в состояние) и **на выходе** (один раз, когда уходит). Например, «вход: глаза зелёные» значит, что глаза загораются зелёным каждый раз, когда Мишка попадает в это состояние, и больше ничего с ними в этом состоянии не происходит.

Как придумать алгоритм управления? Сначала реши, в каких режимах бывает устройство. Потом – что переключает его из режима в режим. И только потом – что оно делает в каждом режиме. Возьмём самое простое устройство – фонарик на лапе.

- Состояния: «Темно» (вход: глаза – выключить) и «Светло» (вход: глаза – белый цвет).
- Переходы: из «Темно» в «Светло» по событию «Кнопка правой лапы: Клик», и обратно по тому же событию.
- Начальное состояние – «Темно».

Кликнули три раза – Мишка прошёл «Темно → Светло → Темно → Светло», глаза горят. Кликнули четыре раза – погасли. Чётное число кликов возвращает в начало. Проверять такие вещи удобнее всего по таблице, о ней – в разделе 7.4.

Три правила, на которых держатся все задачи этой главы:

- машина всегда находится **в одном** состоянии;
- событие, для которого из текущего состояния нет стрелки, **пропадает**: Мишка его как будто не заметил;
- если на стрелке есть условие и оно ложно, перехода нет, и машина остаётся на месте.

Второе правило – источник половины ошибок. Хлопнули в ладоши, пока Мишка спит? Если из «сплю» нет стрелки «громкий звук», хлопок ничего не изменит, даже если в другом состоянии он вызвал бы тревогу.

Таймер – это будильник, и у разных исполнителей он заведён по-разному. У робота «Садовник» (так в курсе) он срабатывает **один раз**: событие «Время вышло» приходит и всё, а чтобы отсчёт пошёл снова, таймер запускают заново – в том же действии, которое выполняется по этому событию. Поэтому «ходить каждые 200 мс» значит: поставить таймер на входе в состояние и перезапустить его по каждому «Времени вышло». У КиберМишки, по справке Кибериады, таймер иначе: команда «Старт» с интервалом 3000 мс даёт событие «Тайм-аут» каждые 3 секунды, пока таймер не остановят командой «Приостановить». Какой у тебя таймер – повторяющийся или одноразовый, смотри в условии задачи или в справке (F1). И в том и в другом случае таймер удобно запускать на входе в состояние и останавливать на выходе из него.

Садовником управляют так же, только события у него другие. Обойти поле вдоль стены можно машиной из двух состояний. В первом Садовник выполняет «Осмотреть стены». Во втором по результату выбирает: если спереди стены нет – «Вперёд», если есть – «Повернуть направо». Потом снова в первое. Такая программа из двух кружков и трёх стрелок поведёт робота вдоль края поля любого размера – круг за кругом, пока его что-нибудь не остановит. Остановку добавляют счётчиком шагов или осмотром цветов: дошёл до клетки, где уже растёт роза, – значит, круг замкнулся.

Запомни. Порядок придумывания: режимы → что их переключает → что делать при входе. Событие без своей стрелки из текущего состояния пропадает. Таймер Садовника срабатывает один раз и запускается заново, таймер КиберМишки (по справке) тикает, пока его не остановят.

Совет. Сдавать схему можно двумя путями. В курсе: открой задачу в задачнике Кибериады, собери диаграмму, проверь её в симуляторе и нажми «Отправить решение». Если проверка пройдена, появится **кодовое слово** – его вводят в поле ответа на сайте курса. На отборе решение, скорее всего, загружают в проверочную систему отборочного тура: как именно, будет написано в инструкции на «Орбите». Перед отправкой в любом случае прогони схему в симуляторе на всех примерах из условия.

7.3. Данные датчика: числа, порог, события

НА ПАЛЬЦАХ



Рисунок 7.3 – Порог – как планка на прыжках в высоту. Событие случается в тот миг, когда сигнал перепрыгнул планку

Пока сигнал под планкой, Мишка молчит. Перепрыгнул – одно событие. Сколько бы сигнал ни висел над планкой потом, второго события не будет, пока он не опустится и не прыгнет снова.

Датчик Мишки меряет не «есть свет – нет света», а **сколько** света. Нос выдаёт яркость числом от 0 до 4095, уши – громкость по той же шкале. Откуда такое странное число? Вспомни главу 4: 12 двоичных разрядов дают $2^{12} = 4096$ разных чисел, от 0 до 4095. Значит, прибор внутри Мишки записывает каждое измерение 12 битами.

Программе нужно понять: это уже «сигнал» или ещё шум? Для этого ставят **порог** – границу, которую значение должно перейти, чтобы Мишка отреагировал. У носа порог настраивают командой «Настроить яркость», у ушей – «Настроить порог громкости». Дальше датчик сам следит за числом и присылает событие, когда оно перешло через порог. Так устроены автоматические двери в магазине и фонарь, который зажигается в сумерках. Ты с этой идеей уже встречался в главе 2.

Считать события по порогу – значит считать **переходы через порог**, а не отсчёты над ним. Если свет продержался над порогом десять измерений подряд, событие «Выше порогового значения» пришло один раз, в первом из них. Событие «Ниже порогового значения» – тоже одно, когда яркость снова упала. В задачах этой книги договоримся: «выше» наступает, когда значение было меньше порога, а стало **не меньше** (больше или равно); «ниже» – когда было не меньше, а стало меньше. Если в задаче на отборе правило задано иначе, работай по условию задачи.

НА ПАЛЬЦАХ

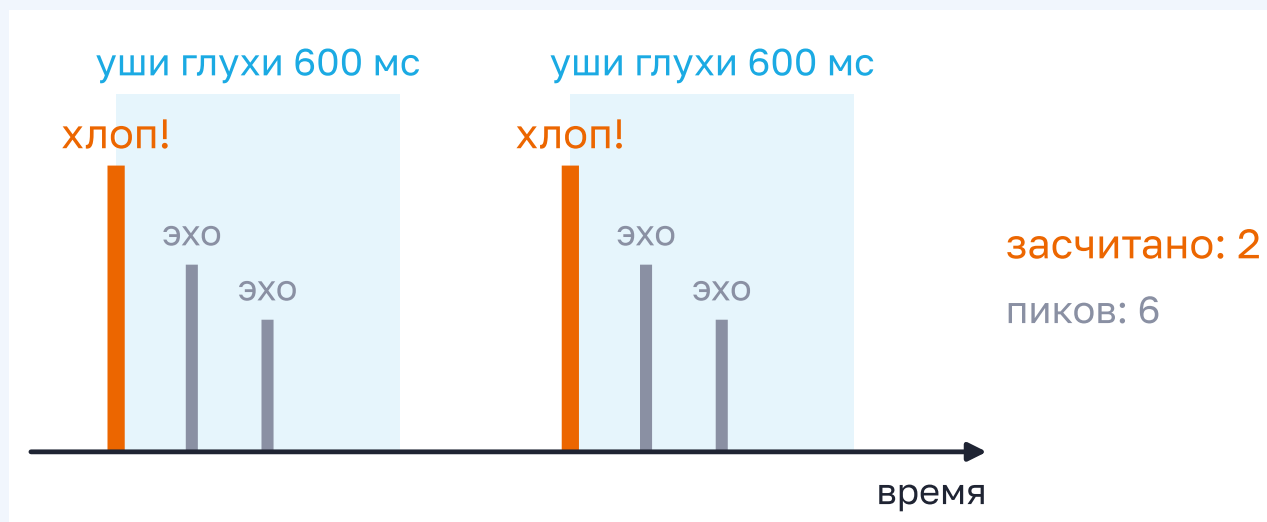


Рисунок 7.4 – Один хлопок – три пика: сам хлопок и два отголоска. После засчитанного хлопка Мишка на время «затыкает уши»

Эхо приходит, пока уши глухи, и Мишка его не замечает. Шесть пиков – два хлопка.

С ушами есть своя беда: **эхо**. Хлопок в ладоши в спортзале звучит не один миг. Звук отражается от стен, и микрофон ловит несколько пиков подряд. Чтобы один хлопок не засчитался трижды, у ушей есть команда «Установить интервал срабатывания»: после того как Мишка отреагировал, он не обращает внимания на громкие звуки заданное число миллисекунд.

Ещё один вид сигнала – **байт**. Компоненты «Уши-байты» и «Нос-байты» принимают сообщение звуком или ИК-лучом (как это устроено, мы разбирали в главе 5) и присылают событие «Получен байт». Сам байт лежит в значении «Полученный байт» – это число от 0 до 255. Что с ним делать, решает программа. Можно положить байт в калькулятор, найти остаток от деления и по остатку выбрать картинку. Можно сравнить байт с числом и по условию на стрелке пойти в одно из состояний.

НА ПАЛЬЦАХ

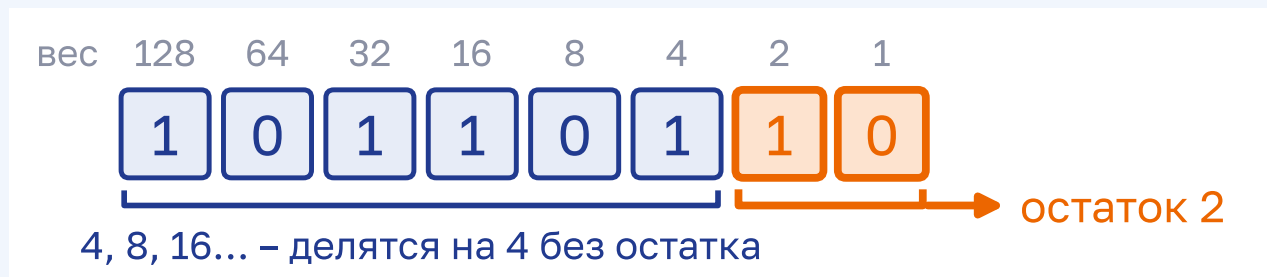


Рисунок 7.5 – Байт 10110110. Левые шесть разрядов делятся на 4 нацело, остаток дают только две правые цифры

Хочешь остаток от деления на 4 – закрой пальцем всё, кроме двух правых цифр.

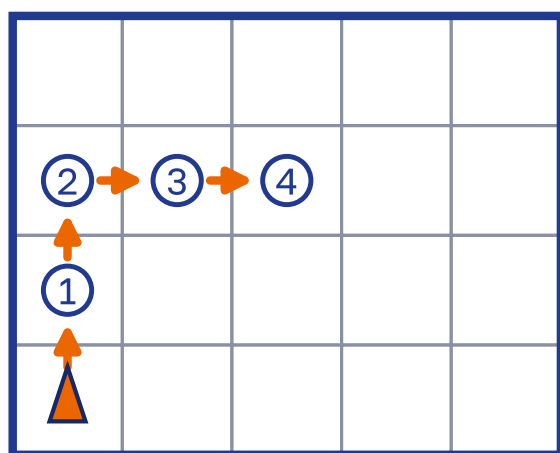
Остаток от деления на 4 полезно уметь находить прямо по двоичной записи. Деление на 2 в двоичной системе – это сдвиг на один разряд вправо, как деление на 10 в десятичной. По-

этому остаток от деления на 4 – это **последние две двоичные цифры**, на 8 – последние три. Пример: 01101110_2 , последние две цифры $10_2 = 2$, значит, остаток от деления на 4 равен 2. Сверим в десятичной: $01101110_2 = 110$, и $110 = 4 \cdot 27 + 2$ – остаток тот же.

Запомни. Шкала датчика 0–4095 – это 12 бит. Событие по порогу – это переход через порог, а не каждый отсчёт над ним. «Интервал срабатывания» глушит эхо. Остаток от деления байта на 4 – две последние двоичные цифры.

7.4. Трассировка: ведём исполнителя пальцем

НА ПАЛЬЦАХ



шаг	команда	где
1	вперёд	(0; 2)
2	вперёд	(0; 1)
–	направо	(0; 1)
3	вперёд	(1; 1)
4	вперёд	(2; 1)

Рисунок 7.6 – Трассировка: после каждой команды записываем, где исполнитель и куда смотрит

Номера в кружках – клетки после каждого шага «вперёд». Поворот клетку не меняет, в таблице у него прочерк вместо номера.

Трассировка – проход по программе шаг за шагом с записью, что происходит после каждого шага. Так программист проверяет чужой код, а на отборе так решают задачи «где окажется исполнитель». Делать это в уме – верный способ ошибиться на пятом шаге. Нужна таблица.

Для Садовника столбцы таблицы такие: номер команды, команда, куда смотрит, x , y . Для машины состояний: время или номер события, событие, в каком состоянии было, есть ли стрелка (и верно ли условие), в каком состоянии стало, какое действие выполнилось. Каждая строка – одна команда или одно событие. Прыгать через строки нельзя, даже если «и так понятно».

Больше всего ошибок у Садовника – на поворотах. «Направо» – это направо **для Садовника**, а не для тебя, сидящего над картинкой. Если он смотрит на юг, то есть к тебе лицом, его правая рука – на западе. Удобно держать в голове круг сторон света по часовой стрелке: С → В → Ю → З → С. «Повернуть направо» – шаг по кругу по часовой стрелке, «налево» – шаг против. Два поворота направо – разворот, три поворота направо – то же, что один налево.

Совет. Если путаешься в поворотах, поверни книгу так, чтобы Садовник на рисунке смотрел от тебя. Тогда его «направо» совпадёт с твоим.

Второй частый вопрос – **сколько команд нужно**. Для Садовника на пустом поле число команд «Вперёд» и «Назад» не может быть меньше, чем клеток по горизонтали плюс клеток по вертикали между стартом и финишем. К ним добавляются повороты: каждая смена направления движения стоит хотя бы одну команду. Поэтому короткая программа – та, в которой меньше поворотов. И не забывай про «Назад»: если финиш сзади, два шага назад дешевле, чем разворот (две команды) и два шага вперёд.

Когда Садовник обходит поле по краю, пригодится подсчёт клеток рамки. У поля шириной W и высотой H клеток по краю $2 \cdot (W + H) - 4$. Почему минус 4? Если сложить две длинные и две короткие стороны, каждая из четырёх угловых клеток посчитается дважды. Для поля 6×4 : $2 \cdot (6 + 4) - 4 = 16$ клеток. Проверь по клеткам на бумаге – по 6 сверху и снизу, и ещё по 2 слева и справа, итого 16.

Запомни. Трассировка – таблица, строка за строкой. Направо – по часовой стрелке $C \rightarrow B \rightarrow \text{Ю} \rightarrow 3$. Меньше команд – значит, меньше поворотов. Клеток в рамке поля $W \times H$: $2 \cdot (W + H) - 4$.

7.5. Разбираем задачи

Задача 7.1 Куда ушёл Садовник

разминка · исполнитель Садовник, трассировка, координаты на поле, повороты

Поле – 6 столбцов и 5 строк, клетки нумеруются с нуля: x – слева направо, y – сверху вниз, север – вверх. Садовник стоит в клетке $(1; 3)$ и смотрит на север. Он выполнил программу:

Вперёд, Вперёд, Повернуть направо, Вперёд, Вперёд, Повернуть направо, Вперёд, Повернуть налево, Вперёд, Назад, Повернуть направо, Вперёд.

В какой клетке он оказался и куда смотрит? В ответ введи x , y и букву стороны света (С, В, Ю или З) через пробел.

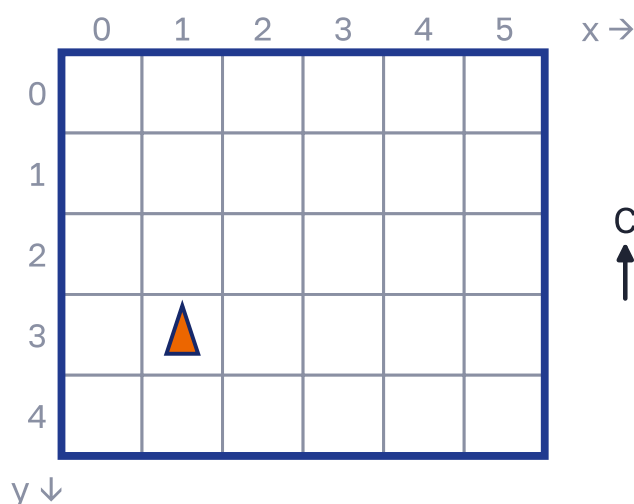


Рисунок 7.7 – Поле 6 × 5. Треугольник – Садовник, остриё показывает, куда он смотрит

Дано: старт (1; 3), смотрит на С; 12 команд; на север y уменьшается, на восток x растёт.
Найти: конечная клетка и направление.

Решение

Шаг 1. Правила движения. Запишем, как каждая команда меняет координаты:

- «Вперёд»: на С – y минус 1, на Ю – y плюс 1, на В – x плюс 1, на З – x минус 1;
- «Назад» – наоборот, а направление не меняется;
- повороты меняют только направление: направо – по кругу С → В → Ю → З, налево – в обратную сторону.

Шаг 2. Трассировка. Ведём таблицу, строка на команду.

Таблица 7.3

№	Команда	Куда смотрит	x	y
–	старт	С	1	3
1	Вперёд	С	1	2
2	Вперёд	С	1	1
3	Направо	В	1	1
4	Вперёд	В	2	1
5	Вперёд	В	3	1
6	Направо	Ю	3	1
7	Вперёд	Ю	3	2
8	Налево	В	3	2
9	Вперёд	В	4	2
10	Назад	В	3	2
11	Направо	Ю	3	2

№	Команда	Куда смотрит	х	у
12	Вперёд	Ю	3	3

Разберём строку 8, она самая коварная. Садовник смотрит на юг, то есть на рисунке – вниз, к тебе. Его левая рука – на востоке. Поэтому «налево» из Ю даёт В: по кругу $C \rightarrow B \rightarrow Ю \rightarrow 3$ шаг назад от Ю – это В.

Шаг 3. Не вышли ли за поле. По ходу таблицы x был от 1 до 4, y – от 1 до 3. Поле – x от 0 до 5, y от 0 до 4. В перегородку Садовник ни разу не шагнул, программа доработала до конца.

Проверим кодом. Строки после `#` – пояснения, Python их не читает.

```
krug = "СВЮЗ" # стороны света по часовой стрелке
shag = {"С": (0, -1), "В": (1, 0), "Ю": (0, 1), "З": (-1, 0)}
x, y, d = 1, 3, "С" # старт
prog = "ВВПВВПВЛВНПВ" # В - вперёд, Н - назад, П - направо, Л - налево
for c in prog:
    dx, dy = shag[d]
    if c == "В": x, y = x + dx, y + dy
    if c == "Н": x, y = x - dx, y - dy
    if c == "П": d = krug[(krug.index(d) + 1) % 4] # шаг по кругу вперёд
    if c == "Л": d = krug[(krug.index(d) - 1) % 4] # шаг по кругу назад
print(x, y, d) # 3 3 Ю
```

Словарь `shag` хранит, на сколько меняются x и y при шаге в каждую сторону. Запись `krug.index(d)` – номер буквы в строке "СВЮЗ", а `% 4` – остаток от деления на 4: после 3 (номер 3) поворот направо даёт номер 4, остаток 0, то есть снова С.

⚠ Типичная ошибка. Шагая на север, прибавить 1 к y , как в тетради по математике. Здесь строки считаются сверху вниз, и север уменьшает y . Не меньше народу спотыкается в строке 8: поворачивают Садовника налево «как себя», на запад. Ещё бывает, что «Назад» понимают как разворот: Садовник при этом **не поворачивается**, он пятится.

Ответ: 3 3 Ю.

Своя задача по программе сферы, модуль 8

Реши сам

7.1. То же поле 6×5 . Садовник стоит в клетке $(0; 0)$ и смотрит на восток. Программа: Вперёд, Вперёд, Вперёд, Повернуть направо, Вперёд, Вперёд, Повернуть направо, Вперёд, Повернуть налево, Вперёд. Где он окажется и куда будет смотреть? В ответ введи x , y и букву стороны света через пробел.

7.2. Поле 6×5 , Садовник в клетке $(2; 1)$ смотрит на север. Ему нужно попасть в клетку $(0; 4)$, в какую сторону он будет смотреть в конце – неважно. Можно пользоваться всеми четырьмя командами движения и поворота. Сколько команд в самой короткой программе?

Задача 7.2 Рамка из роз

исполнитель Садовник, подсчёт команд, клетки по периметру поля

Поле – 7 столбцов и 5 строк, по краям – перегородки. Садовник стоит в северо-западном углу, в клетке $(0; 0)$, и смотрит на восток. Он должен посадить по розе в каждую крайнюю клетку поля, обходя край по часовой стрелке, и остановиться в последней клетке рамки – $(0; 1)$. В каждой клетке рамки Садовник один раз выполняет «Посадить», а поворачивает, только когда впереди перегородка. Команды «Осмотреть стены» не считаем. Сколько роз он посадит и сколько всего команд «Посадить», «Вперёд» и «Повернуть направо» выполнит? В ответ введи два числа через пробел.

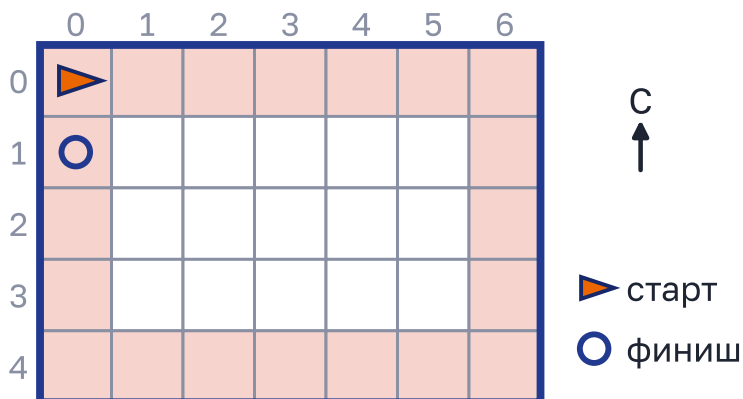


Рисунок 7.8 – Поле 7×5 . Закрашены клетки рамки. Треугольник – старт, кружок – финиш

Дано: поле $W = 7, H = 5$; старт $(0; 0)$, смотрит на В; финиш $(0; 1)$; обход по часовой стрелке.

Найти: число роз; общее число команд.

Решение

Шаг 1. Сколько клеток в рамке. По формуле из раздела 7.4:

$2 \cdot (W + H) - 4 = 2 \cdot (7 + 5) - 4 = 20$. Проверим по сторонам: верхняя строка – 7 клеток, нижняя – 7, в левом и правом столбце между ними – по $5 - 2 = 3$. Итого $7 + 7 + 3 + 3 = 20$. Роз – 20, по одной на клетку, значит, и команд «Посадить» – 20.

Шаг 2. Сколько шагов «Вперёд». Садовник побывает во всех 20 клетках рамки. В первой он уже стоит, в каждую из остальных 19 приходит одним шагом. Шагов – $20 - 1 = 19$.

Шаг 3. Сколько поворотов. Проследим маршрут:

- по верхней строке на восток до клетки $(6; 0)$ – впереди перегородка, поворот направо, теперь смотрит на юг;
- по правому столбцу на юг до $(6; 4)$ – поворот направо, на запад;
- по нижней строке на запад до $(0; 4)$ – поворот направо, на север;
- по левому столбцу на север до $(0; 1)$ – это финиш, дальше клетка $(0; 0)$ уже с розой.

Поворотов – 3. Четвёртый не нужен: Садовник останавливается раньше, чем упрётся в угол.

Шаг 4. Складываем. $20 + 19 + 3 = 42$ команды.

Прикинем другим путём. На каждую клетку – две команды, посадить и шагнуть: $2 \cdot 20 = 40$. Шагов на один меньше, зато добавились три поворота: $40 - 1 + 3 = 42$.

 **Типичная ошибка.** Посчитать клетки рамки как $2 \cdot (7 + 5) = 24$ – угловые клетки попадают в подсчёт дважды. Ещё одна ловушка – решить, что шагов столько же, сколько клеток, 20. В стартовую клетку Садовник не шагает, он в ней начинает. И не добавляй четвёртый поворот «для красоты»: в условии сказано, что поворачивает он, только когда впереди перегородка, а на финише впереди клетка с розой, и программа просто кончается.

Ответ: 20 42.

По мотивам учебной задачи Кибериады IDE «Розовая рамка» (исполнитель Садовник)

Реши сам

7.3. Поле – 9 столбцов и 6 строк. Садовник стоит в $(0; 0)$ и смотрит на юг. Он обходит край поля **против** часовой стрелки, поворачивая налево, только когда впереди перегородка, сажает мяту в каждую крайнюю клетку и останавливается в последней клетке рамки. Сколько кустов мяты он посадит и сколько всего выполнит команд «Посадить», «Вперёд» и «Повернуть налево»? В ответ введи два числа через пробел.

7.4. Задача наоборот. Садовник обошёл по краю квадратное поле так же, как в задаче 7.2, и посадил 36 роз. Сколько клеток в стороне квадрата и сколько всего команд «Посадить», «Вперёд» и «Повернуть направо» он выполнил? В ответ введи два числа через пробел.

Задача 7.3 Мишка-сторож

машина состояний, события кнопки, звука и таймера, трассировка по времени

КиберМишку сделали сторожем по схеме на рисунке. Сначала он в состоянии «Сплю». При входе в «Тревогу» Мишка зажигает красные глаза, пищит и запускает таймер на 3 с, а при выходе из неё останавливает таймер. Все хлопки здесь громкие: каждый громче порога, и уши его слышат. Кликают по кнопке на правой лапе. События по порядку: 1 с – клик, 2 с – хлопок, 3 с – хлопок, 4,5 с – клик, 6 с – хлопок, 9,5 с – клик, 10 с – хлопок.

Сколько раз Мишка поднимет тревогу? В каком состоянии он будет в момент 4,5 с сразу после клика и в момент 12 с? В ответ введи три числа через пробел: число тревог и номера состояний (1 – «Сплю», 2 – «Слушаю», 3 – «Тревога»).

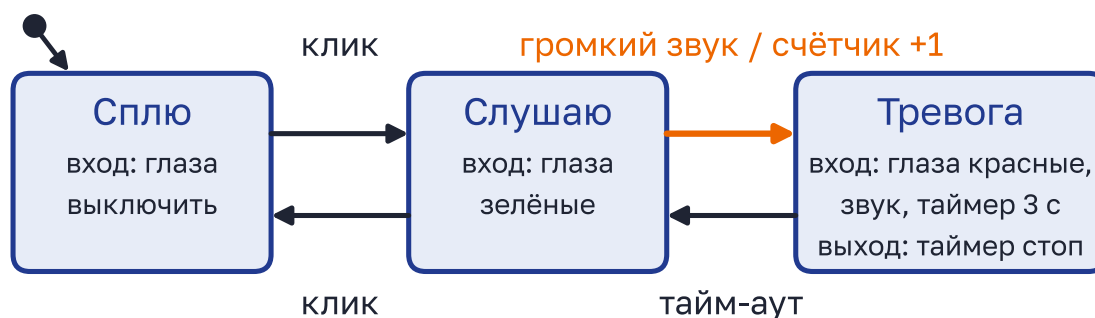


Рисунок 7.9 – Схема Мишки-сторожа. Чёрная точка – начальное состояние

Дано: схема; 7 событий по времени; таймер – 3 с от входа в «Тревогу».

Найти: число тревог; состояние в 4,5 с и в 12 с.

Решение

Шаг 1. Какие стрелки есть. Выпишем из схемы, какое событие что делает в каждом состоянии:

- «Сплю»: клик → «Слушаю»; остальное пропадает;
- «Слушаю»: клик → «Сплю»; громкий звук → «Тревога» и счётчик +1; тайм-аута здесь нет;
- «Тревога»: тайм-аут → «Слушаю»; клика и звука нет – они пропадают.

Шаг 2. Где взять тайм-ауты. Их нет в списке событий – их создаёт таймер. Он запускается при каждом входе в «Тревогу» и через 3 с присылает тайм-аут. Будем добавлять их в таблицу по ходу дела.

Шаг 3. Трассировка.

Таблица 7.4

Время, с	Событие	Было	Есть стрелка?	Стало	Тревог
1	клик	Сплю	да	Слушаю	0
2	хлопок	Слушаю	да	Тревога	1
3	хлопок	Тревога	нет	Тревога	1
4,5	клик	Тревога	нет	Тревога	1
5	тайм-аут	Тревога	да	Слушаю	1
6	хлопок	Слушаю	да	Тревога	2
9	тайм-аут	Тревога	да	Слушаю	2
9,5	клик	Слушаю	да	Сплю	2
10	хлопок	Сплю	нет	Сплю	2

Первый тайм-аут: тревога началась в 2 с, $2 + 3 = 5$ с. Второй: $6 + 3 = 9$ с. Хлопок в 3 с пришёл, когда Мишка уже был в «Тревоге», – стрелки для звука там нет, и второй тревоги этот хлопок не вызвал. Клик в 4,5 с тоже пропал.

Шаг 4. Читаем ответ. Тревог – 2. В 4,5 с Мишка в «Тревоге» (номер 3): клик на него не подействовал. После 10 с событий больше нет, и в 12 с он спит (номер 1).

 **Типичная ошибка.** Считать тревогой каждый хлопок и насчитать 4. Хлопок в «Тревоге» и хлопок во сне стрелки не находят и пропадают. Бывает и наоборот – про тайм-ауты забывают: их нет в списке событий, но без них Мишка «застрянет» в «Тревоге» навсегда, и второй хлопок в 6 с тоже пропадёт.

Ответ: 2 3 1.

Своя задача по программе сферы, модуль 8

Реши сам

7.5. Схема та же, что в задаче 7.3. События: 0,5 с – хлопок, 1 с – клик, 1,5 с – клик, 2 с – хлопок, 3 с – клик, 4 с – хлопок, 6 с – клик, 7,5 с – хлопок, 11 с – хлопок. Сколько раз Мишка поднимет тревогу и в каком состоянии будет в момент 12 с? В ответ введи два числа через пробел (номера состояний – как в задаче 7.3).

7.6.* В схему из задачи 7.3 добавили условие на стрелку из «Слушаю» в «Тревогу»: «громкий звук [счётчик < 2] / счётчик +1». Счётчик в начале равен 0. События: 1 с – клик, 2 с – хлопок, 6 с – хлопок, 8 с – хлопок, 10 с – хлопок, 11 с – клик, 12 с – хлопок. Сколько раз Мишка поднимет тревогу и в каком состоянии будет в момент 13 с? В ответ введи два числа через пробел.

Задача 7.4 Нос и порог

датчик, порог, переходы через порог, длительность

На крыльце дачи Мишка следит носом за инфракрасным фонариком. Порог носа – 2000. Программа такая: по событию «Выше порогового значения» – показать на матрице сердце, по событию «Ниже порогового значения» – очистить матрицу. Нос измеряет яркость каждые 0,2 с, первый отсчёт – в момент 0:

350, 420, 2600, 3100, 2900, 1800, 2100, 2050, 400, 380, 3900, 4000, 1990, 2000, 2400, 600.

«Выше» наступает, когда яркость была меньше 2000, а стала 2000 или больше. «Ниже» – когда была 2000 или больше, а стала меньше. До первого отсчёта считаем, что было темно. Сколько раз загорится сердце и сколько секунд всего оно будет гореть за эти 3 секунды? В ответ введи два числа через пробел.

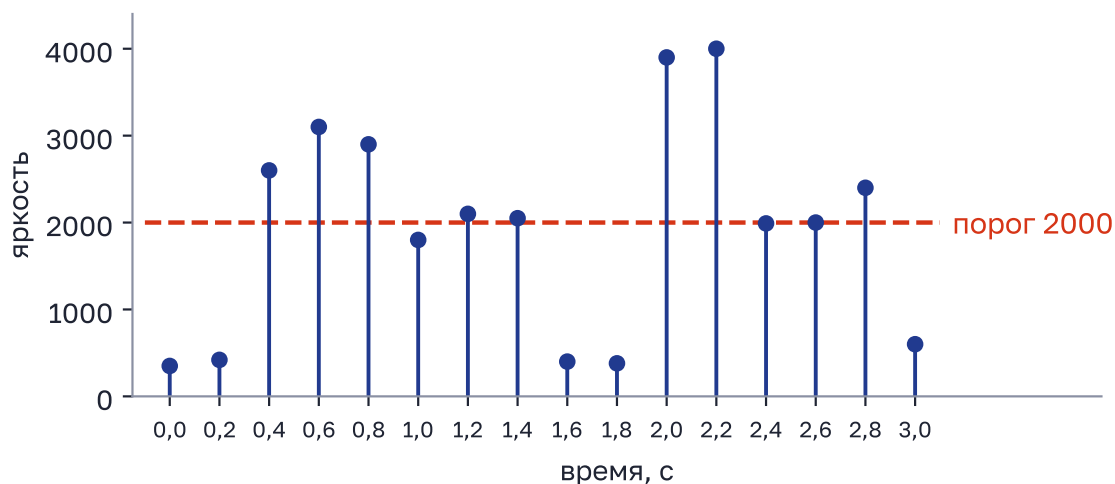


Рисунок 7.10 – Отсчёты яркости и порог 2000

Дано: 16 отсчётов через 0,2 с; порог 2000; «выше» – переход снизу на 2000 и больше.

Найти: сколько раз загорится сердце; общее время горения, с.

Решение

Шаг 1. Отметим каждый отсчёт: над порогом (Н) или под ним (П). Время отсчёта – его номер, умноженный на 0,2 с, считая с нуля.

Таблица 7.5

№	Время, с	Яркость	Где
0	0	350	П
1	0,2	420	П
2	0,4	2600	Н
3	0,6	3100	Н
4	0,8	2900	Н
5	1,0	1800	П
6	1,2	2100	Н
7	1,4	2050	Н
8	1,6	400	П
9	1,8	380	П
10	2,0	3900	Н
11	2,2	4000	Н
12	2,4	1990	П
13	2,6	2000	Н
14	2,8	2400	Н
15	3,0	600	П

Отсчёт 13 – 2000, точно на пороге, а по условию «2000 или больше» он уже над ним. А 1990 в отсчёте 12 – под порогом, хоть и совсем чуть-чуть.

Шаг 2. Ищем смены П → Н (сердце загорается) и Н → П (гаснет).

- П → Н в 0,4 с, Н → П в 1,0 с: горело $1,0 - 0,4 = 0,6$ с;
- П → Н в 1,2 с, Н → П в 1,6 с: 0,4 с;
- П → Н в 2,0 с, Н → П в 2,4 с: 0,4 с;
- П → Н в 2,6 с, Н → П в 3,0 с: 0,4 с.

Сердце загорелось 4 раза.

Шаг 3. Общее время. $0,6 + 0,4 + 0,4 + 0,4 = 1,8$ с. Из трёх записанных секунд сердце горело почти две. На графике над красной линией 9 палочек из 16 – тоже больше половины.

Проверим кодом.

```
v = [350, 420, 2600, 3100, 2900, 1800, 2100, 2050, 400, 380, 3900, 4000, 1990, 2000, 2400, 600]
porog = 2000
bylo = False          # до первого отсчёта темно
raz, gorit = 0, 0      # сколько раз загорелось, сколько отсчётов горело
for x in v:             # x – яркость очередного отсчёта
    stalo = x >= porog
    if stalo and not bylo: raz += 1      # переход П → Н
    if bylo: gorit += 1                 # горело весь промежуток до этого отсчёта
    bylo = stalo
print(raz, round(gorit * 0.2, 1))      # 4 1.8
```

Переменная `bylo` помнит, был ли предыдущий отсчёт над порогом. Если был, значит, последнее 0,2 с сердце горело, и мы добавляем один промежуток. Так считается время без ручного вычитания.

⚠ Типичная ошибка. Посчитать, сколько отсчётов над порогом, и ответить 9. Сердце реагирует на переходы, а не на каждый отсчёт. Другая ловушка – решить, что 2000 при пороге 2000 «ещё не выше». Тогда четвёртый раз сердце загорится только в 2,8 с, и время выйдет 1,6 с вместо 1,8.

Ответ: 4 1,8.

Своя задача по программе сферы, модуль 9

Реши сам

7.7. Мишка считает хлопки ушами, интервал срабатывания – 600 мс: хлопок засчитывается, только если с последнего засчитанного прошло не меньше 600 мс. Первый хлопок засчитывается всегда. Громкие звуки пришли в моменты 0, 250, 700, 1000, 1500, 1650, 2300, 2500, 2900 мс. Сколько хлопков засчитал Мишка?

7.8. Нос Мишки получил 13 отсчётов: 500, 1500, 2500, 3500, 2600, 1200, 900, 2200, 3300, 1100, 3100, 2400, 800. Правила «выше» и «ниже» – как в задаче 7.4. Мишка может поставить порог 1000, 2000 или 3000. При каком пороге событие «Выше порогового значения» случится 2 раза?

Задача 7.5 Картинки по звуку

приём байтов, двоичная запись, остаток от деления, реакция на сигнал

Мишка принимает байты ушами. По событию «Получен байт» программа кладёт байт в калькулятор, находит остаток от деления на 4 и показывает картинку по таблице:

Таблица 7.6

Остаток	0	1	2	3
Картинка	сердце	улыбка	грусть	нота

Пришли пять байтов: 10110110, 00011011, 11100000, 01010101, 10000001. Какие остатки получит калькулятор? Введи пять цифр подряд, без пробелов.

Дано: 5 байтов в двоичной записи; остаток от деления на 4 → картинка.

Найти: пять остатков по порядку.

Решение

Шаг 1. Способ первый: перевести в десятичную и поделить. Вес разрядов байта справа на лево: 1, 2, 4, 8, 16, 32, 64, 128. Складываем веса тех разрядов, где стоит 1.

- $10110110_2 = 128 + 32 + 16 + 4 + 2 = 182; 182 = 4 \cdot 45 + 2$, остаток 2;
- $00011011_2 = 16 + 8 + 2 + 1 = 27; 27 = 4 \cdot 6 + 3$, остаток 3;
- $11100000_2 = 128 + 64 + 32 = 224; 224 = 4 \cdot 56$, остаток 0;
- $01010101_2 = 64 + 16 + 4 + 1 = 85; 85 = 4 \cdot 21 + 1$, остаток 1;
- $10000001_2 = 128 + 1 = 129; 129 = 4 \cdot 32 + 1$, остаток 1.

Шаг 2. Способ второй: смотреть на две последние цифры. Все разряды, кроме двух правых, весят 4, 8, 16 и больше – они делятся на 4 без остатка. Остаток дают только два последних разряда с весами 2 и 1.

Таблица 7.7

Байт	Две последние цифры	Остаток	Картинка
10110110	10	2	грусть
00011011	11	3	нота
11100000	00	0	сердце
01010101	01	1	улыбка
10000001	01	1	улыбка

Оба способа дали одно и то же, а второй занял секунды. На отборе, где время дорого, считай вторым, а первым проверяй.

Шаг 3. Что увидим в конце. Последним пришёл 10000001, на животе у Мишки останется улыбка.

 **Типичная ошибка.** Смотреть на **первые** две цифры байта вместо последних. Для 10110110 выйдет 10 – случайно тот же остаток 2, и ученик решает, что способ верный. Но для 11100000 первые цифры дадут 3 вместо 0. А при переводе в десятичную не начинай веса с 1 слева: единица – вес самого правого разряда.

Ответ: 23011.

Своя задача по программе сферы, модуль 9

Реши сам

7.9. Программа из задачи 7.5 получила байты, которые записаны десятичными числами: 45, 130, 77, 200, 3. Какие остатки получит калькулятор? Введи пять цифр подряд.

7.10. Программу из задачи 7.5 переделали: калькулятор находит остаток от деления байта на 8, и по остатку от 0 до 7 глаза загораются одним из восьми цветов. Пришли байты 11010110, 01111111, 10101000, 00110011. Какие остатки получит калькулятор? Введи четыре цифры подряд, без пробелов.

Задача 7.6 Строчник ищет кота

исполнитель Строчник, распознавание слова, машина состояний, трассировка

Строчник читает строку по одной букве и должен подавать импульс А каждый раз, когда дочитал слово КОТ. Его схема – на рисунке. Правила по состояниям:

- «Жду К» (начальное): буква К → «Было К»; любая другая → остаёмся в «Жду К»;
- «Было К»: О → «Было КО»; К → остаёмся в «Было К»; любая другая → «Жду К»;
- «Было КО»: Т → импульс А и «Жду К»; К → «Было К»; любая другая → «Жду К».

Строка: **ККОТКОКОТОКОТК**. Сколько импульсов А подаст Строчник и в каком состоянии будет после последней буквы? В ответ введи два числа через пробел, состояние – номером: 1 – «Жду К», 2 – «Было К», 3 – «Было КО».

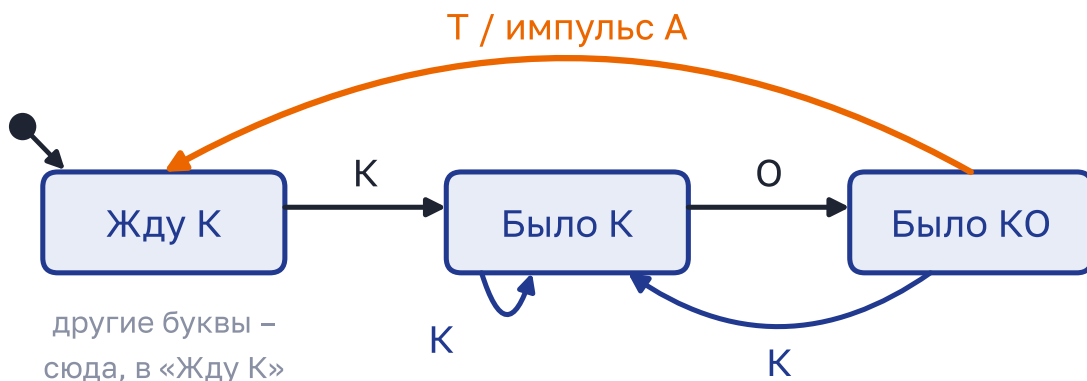


Рисунок 7.11 – Схема Строчника. Синие стрелки по букве К не дают потерять начало слова

Дано: схема из трёх состояний; строка из 14 букв.

Найти: число импульсов А; состояние в конце.

Решение

Шаг 1. Что помнит каждое состояние. Строчник не видит строку целиком, он помнит только своё состояние. «Жду К» – «пока ничего похожего на кота». «Было К» – «последняя буква К, может начаться слово». «Было КО» – «последние две буквы К и О, осталось Т». Отсюда и синие стрелки: если в «Было КО» пришла К, начало нового слова уже есть, и надо идти в «Было К», а не в самое начало.

Шаг 2. Трассировка.

Таблица 7.8

№	Буква	Было	Стало	Импульс
1	К	Жду К	Было К	
2	К	Было К	Было К	
3	О	Было К	Было КО	
4	Т	Было КО	Жду К	1
5	К	Жду К	Было К	
6	О	Было К	Было КО	
7	К	Было КО	Было К	
8	О	Было К	Было КО	
9	Т	Было КО	Жду К	2
10	О	Жду К	Жду К	
11	К	Жду К	Было К	
12	О	Было К	Было КО	
13	Т	Было КО	Жду К	3
14	К	Жду К	Было К	

Шаг 3. Итог и проверка глазами. Импульсов три. После последней К Строчник в «Было К» (номер 2) – если бы строка продолжилась буквами ОТ, получился бы четвёртый кот. Теперь посмотри на саму строку: **ККОТКОКОТОКОТК** – слово КОТ встречается три раза, столько же и импульсов.

Проверим кодом. Машину состояний легко записать словарём: для каждой пары «состояние, буква» – куда идти.

```
kuda = {("Жду К", "К"): "Было К",
        ("Было К", "О"): "Было КО", ("Было К", "К"): "Было К",
        ("Было КО", "К"): "Было К"}
sost, impulsy = "Жду К", 0
for буква in "ККОТКОКОТОКОТК":
    if sost == "Было КО" and буква == "Т":
        impulsy += 1 # дочитали КОТ – импульс А
    sost = kuda.get((sost, буква), "Жду К") # нет пары в словаре – «иначе»
print(impulsy, sost) # 3 Было К
```

Команда `kuda.get(ключ, "Жду К")` ищет пару в словаре, а если её там нет, возвращает «Жду К». Так в коде записана стрелка «любая другая буква». Переход по Т из «Было КО» тоже ведёт в «Жду К», поэтому отдельно его в словарь не записываем.

❗ Типичная ошибка. Нарисовать схему без синих стрелок: «любая неправильная буква – в самое начало». Тогда вторая К в строке 2 сбросит Строчника в «Жду К», первый кот (буквы 2–4) потеряется, и второй (буквы 7–9) тоже. Такая схема насчитает одного кота вместо трёх. Если решаешь по готовой схеме, как здесь, ошибка другая: пропустить в трассировке строку с «лишней» буквой, например № 10, и сбиться со счёта.

Ответ: 3 2.

По мотивам учебной задачи Кибериады IDE «Строчник и слова»

Реши сам

7.11. Схема – как в задаче 7.6. Строчник читает строку **СКОТКОКОТИКОТ**. Сколько импульсов А он подаст и в каком состоянии будет в конце? В ответ введи два числа через пробел.

7.12.* Ученик нарисовал схему для слова КОТ без синих стрелок: в «Было К» и «Было КО» любая буква, кроме нужной, отправляет Строчника в «Жду К». Сколько импульсов подаст правильная схема из задачи 7.6 и сколько – схема ученика на строке **ККОТКОКОТ**? В ответ введи два числа через пробел.

Задача 7.7* Кто ответит на клик

иерархическая машина состояний, передача события внешнему состоянию, трассировка

Задача на вложенные состояния из раздела 6.4: одно состояние лежит внутри другого, как комната в доме. У Мишки внутри состояния «Работа» лежат два вложенных – «Слушаю» (на-

чальное) и «Показываю». Отдельно есть состояние «Сон». Стрелки:

- «Слушаю» → «Показываю» – по событию «байт», при входе в «Показываю» на матрице появляется картинка;
- «Показываю» → «Слушаю» – по клику;
- «Работа» → «Сон» – по клику и по событию «переворот» (положение изменилось);
- «Сон» → «Работа» – по клику, при этом Мишка попадает в начальное «Слушаю».

Правило вложенных состояний: событие сначала получает вложенное состояние. Если у него нет стрелки для этого события, событие передаётся внешнему «Работа». Если нет и у него – событие пропадает. Мишка стартует в «Слушаю». События по порядку:

байт, клик, клик, байт, клик, байт, байт, переворот, клик, байт, клик.

Сколько раз на матрице появится картинка, сколько раз Мишка уснёт и в каком состоянии окажется в конце? В ответ введи три числа через пробел, конечное состояние – номером: 1 – «Сон», 2 – «Слушаю», 3 – «Показываю».

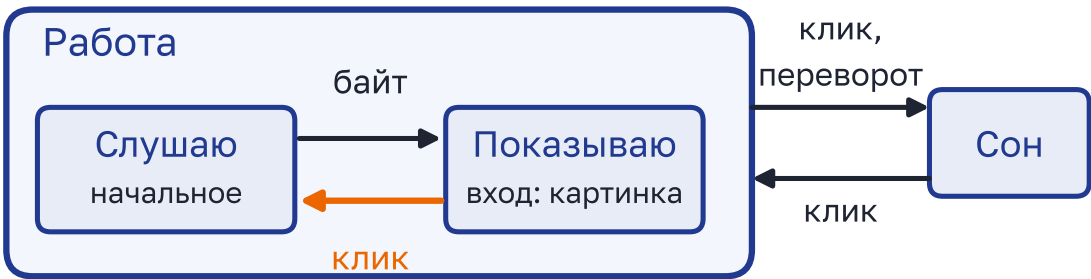


Рисунок 7.12 – Схема с вложенными состояниями. Стрелки от рамки «Работа» действуют из любого вложенного состояния

Дано: «Работа» = {«Слушаю», «Показываю»} и «Сон»; 11 событий; старт в «Слушаю».

Найти: число картинок; число засыпаний; конечное состояние.

Решение

Шаг 1. Кто отвечает на какое событие. Сведём стрелки в таблицу: что происходит с каждым событием в каждом состоянии. Там, где у вложенного состояния своей стрелки нет, смотрим на «Работу».

Таблица 7.9

Состояние	байт	клик	переворот
Слушаю	→ Показываю	«Работа»: → Сон	«Работа»: → Сон
Показываю	пропадает	→ Слушаю	«Работа»: → Сон
Сон	пропадает	→ Слушаю	пропадает

Главная клетка таблицы – клик в «Показываю». У «Показываю» есть своя стрелка на клик, поэтому до «Работы» клик не доходит, и Мишка возвращается в «Слушаю», а не засыпает. Байт в «Показываю» пропадает: своей стрелки нет, у «Работы» тоже.

Шаг 2. Трассировка.

Таблица 7.10

№	Событие	Было	Кто ответил	Стало
1	байт	Слушаю	Слушаю	Показываю, картинка 1
2	клик	Показываю	Показываю	Слушаю
3	клик	Слушаю	Работа	Сон, засыпание 1
4	байт	Сон	никто	Сон
5	клик	Сон	Сон	Слушаю
6	байт	Слушаю	Слушаю	Показываю, картинка 2
7	байт	Показываю	никто	Показываю
8	переворот	Показываю	Работа	Сон, засыпание 2
9	клик	Сон	Сон	Слушаю
10	байт	Слушаю	Слушаю	Показываю, картинка 3
11	клик	Показываю	Показываю	Слушаю

Шаг 3. Итог. Картинка появлялась 3 раза, Мишка уснул 2 раза, в конце он в «Слушаю» (номер 2). Быстрая проверка картинок: они появляются только от байта, пришедшего в «Слушаю», – это события № 1, 6 и 10, их три.

 **Типичная ошибка.** Отдать клик в «Показываю» внешнему состоянию и отправить Мишку спать: тогда засыпаний выйдет три, и закончит он во сне. Вложенное состояние всегда спрашивают первым. Ещё ошибаются так: после «Сна» возвращают Мишку в то вложенное состояние, из которого он уснул (в строке 9 – в «Показываю»). По условию вход в «Работу» всегда ведёт в начальное «Слушаю».

Ответ: 3 2 2.

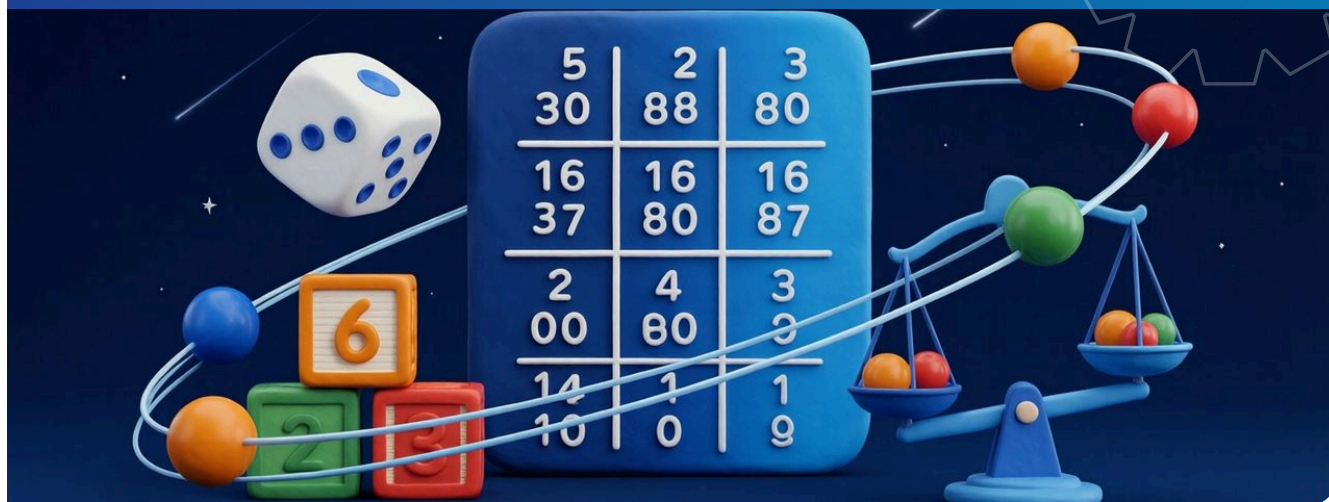
По мотивам задачи второго отборочного этапа НТО 2025/26, профиль «Интеллектуальные энергетические системы», «Киберфизика по Маркову»

Реши сам

7.13.* Схема та же, что в задаче 7.7, Мишка стартует в «Слушаю». События: клик, байт, байт, клик, байт, переворот, переворот, клик, клик, клик, байт. Сколько раз появится картинка, сколько раз Мишка уснёт и в каком состоянии окажется? В ответ введи три числа через пробел.

7.14.* В схему из задачи 7.7 добавили стрелку «Показываю» → «Слушаю» по событию «переворот». Мишка стартует в «Слушаю». События: байт, переворот, байт, клик, переворот, клик, байт, переворот. Сколько раз Мишка уснёт и в каком состоянии окажется в конце? В ответ введи два числа через пробел.

Математика киберфизика



Здесь собрана школьная математика, без которой не решить задачи остальных глав: формулы, единицы, проценты, уравнения, логика, подсчёт вариантов и электронные таблицы. Все примеры – про сигналы и связь. В задачах сферы математика прячется внутри истории: в условии передатчик и тоны, а решать надо уравнение. Споткнулся в другой главе о проценты или килогерцы – загляни сюда.

8.1. Формула – это фраза

НА ПАЛЬЦАХ



Рисунок 8.1 – Формула – как машинка с окошками: вставил числа, получил ответ

В окошки кладём частоту тона и длительность бита. Машинка умножает и выдаёт, сколько колебаний тона уместилось в один бит.

Формула – правило расчёта, записанное коротко. Буква в ней обозначает величину, знаки – действия. Прочитать формулу – значит сказать её словами. Вот формула из главы 5:

$$N = f \cdot \tau.$$

Словами: **число колебаний равно частоте, умноженной на длительность**. Здесь N – сколько колебаний тона уместилось в бит (штук), f – частота тона в герцах, то есть сколько колебаний он делает за секунду, τ («тау») – сколько секунд длится бит. Откуда она взялась? Тон делает 2000 колебаний каждую секунду. За половину секунды он успеет вдвое меньше, 1000, а за 0,05 с – $2000 \cdot 0,05 = 100$ колебаний.

Когда подставляешь числа, иди всегда по одним и тем же шагам – тогда ошибок почти не бывает:

- выпиши все величины из условия вместе с единицами;
- переведи их в те единицы, в которых записана формула: частоту – в герцы, время – в секунды, объём – в биты;
- подставь числа вместо букв;
- посчитай и допиши единицу к ответу.

Порядок действий в формуле такой же, как в школьном примере: сначала скобки, потом умножение и деление, потом сложение и вычитание. Черта дроби – это деление, и работает она как скобки: всё, что над чертой, считают целиком, и только потом делят. Пакет данных состоит из 16 служебных бит и n байт сообщения, а идёт по каналу со скоростью v . Время его передачи

$$t = \frac{16 + 8 \cdot n}{v}.$$

Пусть $n = 5$ байт, $v = 40$ бит/с. Сначала верх: $8 \cdot 5 = 40$, $16 + 40 = 56$ бит. Потом делим: $56 : 40 = 1,4$ с. Кто сложит $16 + 8 = 24$ и умножит на 5, получит 120 бит и 3 с вместо 1,4 – больше чем вдвое.

Одна формула – три задачи. Если в формуле три буквы, по любым двум находится третья. Помогает треугольник: наверх пишем то, что получается умножением, вниз – два множителя. Закрой пальцем то, что ищешь. Остались две буквы рядом – перемножь их. Остались одна над другой – верхнюю раздели на нижнюю.

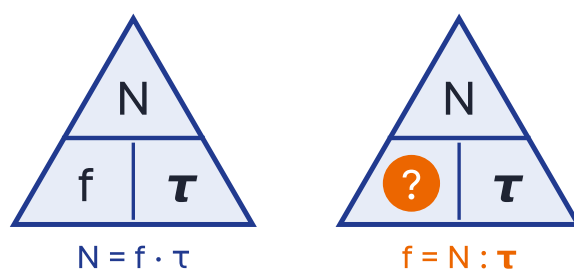


Рисунок 8.2 – Треугольник формулы $N = f \cdot \tau$. Закрыв f – осталось N над τ , значит, $f = N : \tau$

Так же устроены и другие формулы книги: объём сообщения $I = v \cdot t$ (бит = скорость · время) и связь частоты с периодом $f \cdot T = 1$.

Ещё одна привычка сэкономит много баллов: с единицами обращайся как с числами. Герц – «раз в секунду», умножаем на секунды – остаются просто разы, штуки:

$2000 \text{ Гц} \cdot 0,05 \text{ с} = 100$. Биты делим на биты в секунду – остаются секунды. Если единица ответа вышла странной, например «бит · с», формула перевернута.

Запомни. Сначала скажи формулу словами, потом подставляй. Числа – в тех единицах, в которых записана формула: герцы, секунды, биты. Черта дроби делит и одновременно работает как скобки.

8.2. Единицы: герцы, секунды, биты

НА ПАЛЬЦАХ

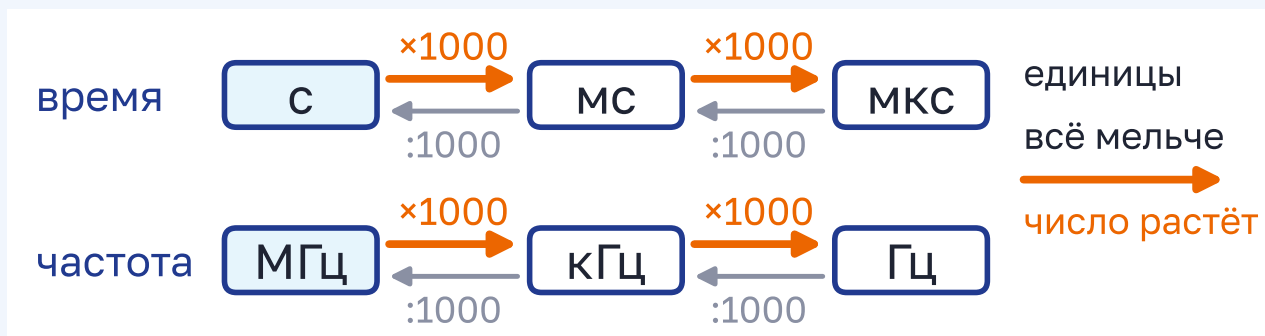


Рисунок 8.3 – Цепочки единиц: вправо, к более мелкой единице, – умножаем на 1000, влево – делим

Шагнул к мелкой единице – кусочки стали меньше, и их понадобилось больше. Поэтому вправо по цепочке число растёт.

В задачах про сигналы три семейства единиц. У каждого одни и те же **приставки**:

- **кило-** (к) – тысяча: $1 \text{ кГц} = 1000 \text{ Гц}$, $1 \text{ кбит/с} = 1000 \text{ бит/с}$;
- **мега-** (М) – миллион: $1 \text{ МГц} = 1000 \text{ кГц} = 1\,000\,000 \text{ Гц}$;
- **милли-** (м) – тысячная часть: $1 \text{ мс} = 0,001 \text{ с}$;
- **микро-** (мк) – миллионная часть: $1 \text{ мкс} = 0,001 \text{ мс} = 0,000\,001 \text{ с}$.

Частоту меряют в **герцах**: 1 Гц – одно колебание в секунду. Время – в секундах, миллисекундах, микросекундах. Объем сообщения – в **битах** (одна двоичная цифра) и **байтах** (8 бит), скорость связи – в битах в секунду.

Правило перевода одно: **к более мелкой единице – умножаем, к более крупной – делим**. Проверь себя смыслом: миллисекунд в секунде много, значит, в миллисекундах число должно выйти больше.

- $38 \text{ кГц} = 38 \cdot 1000 = 38\,000 \text{ Гц}$. С такой частотой мигает свет у многих ИК-пультов.
- $0,5 \text{ мс} = 0,5 : 1000 = 0,0005 \text{ с}$.
- $250 \text{ мкс} = 250 : 1000 = 0,25 \text{ мс}$.
- $2,4 \text{ кбит/с} = 2,4 \cdot 1000 = 2400 \text{ бит/с}$.
- $3 \text{ Кбайт} = 3 \cdot 1024 = 3072 \text{ байта} = 3072 \cdot 8 = 24\,576 \text{ бит}$.

Последний пример с подвохом. Объем памяти в школьной информатике считают по 1024: 1 Кбайт = 1024 байта. Частоту и скорость связи – по 1000: $1 \text{ кбит/с} = 1000 \text{ бит/с}$. Если в условии сказано по-другому, решай по условию.

Частота и период. Период T – время одного колебания, частота f – сколько колебаний в секунду. Они связаны формулой $T = 1 : f$. У этой пары приставки «сокращаются» удобно: 1 кГц даёт период 1 мс, 1 МГц – 1 мкс. Значит, частоту в килогерцах можно сразу делить на единицу и получать миллисекунды: 4 кГц – это $1 : 4 = 0,25$ мс. По той же причине миллисекунды, умноженные на килогерцы, сразу дают штуки: $50 \text{ мс} \cdot 2 \text{ кГц} = 100$ колебаний. Тысяча в «кило» и тысячная в «милли» гасят друг друга.

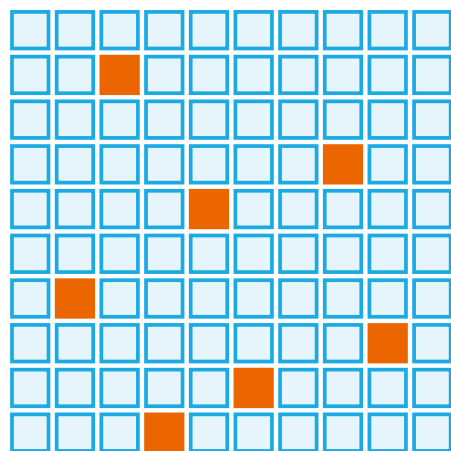
Запомни. К более мелкой единице – умножаем, к более крупной – делим. Кило – тысяча, мега – миллион, милли – тысячная, микро – миллионная. 1 байт = 8 бит. Связь – по 1000 (кбит/с), память – по 1024 (Кбайт).

Совет. Пиши единицу рядом с каждым числом, даже в черновике. Встретились в одном действии килогерцы и секунды или байты и бит/с – остановись и переведи, пока не начал считать.

8.3. Доли, проценты, пропорции

НА ПАЛЬЦАХ

отправили 100 пакетов



□ дошёл
■ потерялся

7 из 100 = 7 %

один процент – одна клетка из ста

Рисунок 8.4 – Сто пакетов, семь из них потерялись по дороге

Процент – одна клетка из ста. Потерялись 7 пакетов из 100 – потери 7 %.

Доля – часть от целого, записанная дробью. Из 8 бит байта испортились 3 – это $\frac{3}{8}$ байта. Дробь – это деление: $\frac{3}{8} = 3 : 8 = 0,375$. **Процент** – сотая часть, поэтому долю переводят в проценты умножением на 100: $0,375 \cdot 100 = 37,5$ %. Обратно – делением: $4 \% = 4 : 100 = 0,04$.

С процентами делают три вещи.

- **Процент от числа.** 5 % от 400 приёмников: $400 \cdot 5 : 100 = 20$ приёмников.

- **Какую долю одно число составляет от другого.** Из 240 бит испортились 18:
 $18:240 \cdot 100 = 7,5 \%$.
- **Увеличить или уменьшить на процент.** Плюс 20 % – умножить на $1 + 0,2 = 1,2$, минус 20 % – умножить на $1 - 0,2 = 0,8$.

«На сколько процентов изменилось» считают от того, что было **до** изменения. Скорость канала выросла с 200 до 250 бит/с: $(250 - 200):200 \cdot 100 = 25 \%$. Обратно, с 250 до 200 – это уже падение на $50:250 \cdot 100 = 20 \%$: теперь сравниваем с 250.

Отдельная путаница бывает, когда меняются сами проценты. Потери в канале были 3 %, после ремонта антенны стали 1,2 %. Можно сказать «потери уменьшились на 1,8» – это разница двух процентов, её называют **процентными пунктами**. А можно сказать «уменьшились на 60 %» – это мы сравнили 1,2 с тремя: $(3 - 1,2):3 \cdot 100 = 60 \%$. Оба ответа честные, но отвечают на разные вопросы. Перечитай, что спрашивают.

Пропорция. Две величины **прямо пропорциональны**, если во сколько раз растёт одна, во столько же раз растёт и другая. Канал за 4 с передал 60 бит. За 10 с – в $10:4 = 2,5$ раза дольше – передаст в 2,5 раза больше: $60 \cdot 2,5 = 150$ бит. Можно и через одну секунду: $60:4 = 15$ бит за секунду, за 10 с – 150 бит.

Обратная пропорциональность – одна величина растёт, другая во столько же раз уменьшается. Файл при скорости 100 бит/с идёт 12 с. При скорости 300 бит/с – в 3 раза быстрее, за 4 с. Бит в файле столько же, просто каждый проходит втрое быстрее.

Деление в отношении. «Приёмники первого, второго и третьего заводов – в отношении 3 : 1 : 6» значит: всё количество делят на $3 + 1 + 6 = 10$ равных частей, первому заводу три части, второму одна, третьему шесть. Если приёмников 400, одна часть – $400:10 = 40$ штук.

Запомни. Долю в проценты – умножить на 100. «На сколько процентов» – от того, что было до. Прямая пропорция: больше времени – больше бит во столько же раз. Обратная: скорость выше в 3 раза – время меньше в 3 раза.

8.4. Уравнения и системы уравнений

НА ПАЛЬЦАХ



Рисунок 8.5 – Весы в равновесии: коробка с неизвестным числом бит и гирька 16 бит против 176 бит

Сними гирьку 16 с левой чаши и столько же с правой – весы останутся в равновесии. В коробке 160 бит.

Уравнение – равенство, в котором одно число неизвестно. Его обозначают буквой, например n или x . Решить уравнение – найти такое число, при котором равенство верно. Решают его как весы в равновесии: с обеими частями делают одно и то же, и равенство не ломается. В простых уравнениях хватает **обратных действий**: к неизвестному прибавили 16 – вычитаем 16 из обеих частей; неизвестное умножили на 8 – обе части делим на 8.

Вернёмся к пакету из раздела 8.1. Пакет на скорости 40 бит/с шёл 1,4 с. Сколько в нём байт сообщения? Пишем формулу и подставляем, что знаем:

$$\frac{16 + 8n}{40} = 1,4.$$

- Умножаем обе части на 40 (черта дроби была делением на 40): $16 + 8n = 56$.
- Вычитаем 16: $8n = 40$.
- Делим на 8: $n = 5$ байт.

Проверка подстановкой – обязательный последний шаг: $(16 + 8 \cdot 5) : 40 = 56 : 40 = 1,4$ с. Совпало с условием – значит, $n = 5$.

Подбор. Бывает, что неизвестное стоит в неудобном месте – например, в степени. Сколько бит n нужно, чтобы 2^n было не меньше 50 (глава 4)? Пробуем по очереди: $2^5 = 32$ – мало, $2^6 = 64$ – хватает. $n = 6$. Подбор – честный способ, если ответ целый и небольшой. Выписывай пробы столбиком, чтобы не пропустить ни одной.

НА ПАЛЬЦАХ

первое: 500 мс

К

Д

К

второе: 400 мс

К

Д

лишний К:
 $500 - 400 = 100$ мс

Рисунок 8.6 – Два сообщения из одинаковых тонов. Одинаковые куски стоят друг под другом, лишним остался один короткий тон

Первое сообщение длиннее второго на один короткий тон и на 100 мс. Значит, короткий тон длится 100 мс.

Система уравнений – когда неизвестных два и про них известны два факта. Одного уравнения с двумя неизвестными не хватает: у него много решений. Допустим, сообщение передают короткими и длинными тонами. Сообщение «2 коротких и 1 длинный» заняло 500 мс, сообщение «1 короткий и 1 длинный» – 400 мс. Сколько длится каждый тон? Пусть короткий тон – k мс, длинный – d мс:

$$\begin{cases} 2k + d = 500, \\ k + d = 400. \end{cases}$$

Способ 1 – подбор. Берём k и из второго уравнения находим d , потом проверяем первое. $k = 50$: $d = 350$, $2 \cdot 50 + 350 = 450$ – не 500. $k = 100$: $d = 300$, $2 \cdot 100 + 300 = 500$ – подходит.

Способ 2 – вычитание. Сравни два сообщения. Первое длиннее второго на один короткий тон, и по времени оно длиннее на $500 - 400 = 100$ мс. Значит, $k = 100$ мс. На языке уравнений: из левой части первого вычли левую часть второго, из правой – правую:

$(2k + d) - (k + d) = 500 - 400$, и длинный тон исчез. Дальше $d = 400 - 100 = 300$ мс.

Способ 3 – сложение. Им пользуются, когда неизвестное стоит в уравнениях с разными знаками. Два тона частотной модуляции (глава 5) в сумме дают 3000 Гц, а разность между высоким и низким – 1000 Гц:

$$\begin{cases} f_1 + f_0 = 3000, \\ f_1 - f_0 = 1000. \end{cases}$$

Складываем левые части с левыми, правые с правыми: f_0 и $-f_0$ взаимно уничтожаются, остаётся $2f_1 = 4000$, $f_1 = 2000$ Гц. Тогда $f_0 = 3000 - 2000 = 1000$ Гц. Это правило из справочника «большее – это сумма плюс разность, делённая пополам», только записанное уравнениями.

Запомни. С обеими частями уравнения делай одно и то же. Два неизвестных – нужны два уравнения. Одинаковые знаки у неизвестного – вычитай уравнения, разные – складывай. Ответ всегда проверяй подстановкой в оба уравнения.

8.5. Логика: И, ИЛИ, НЕ

НА ПАЛЬЦАХ

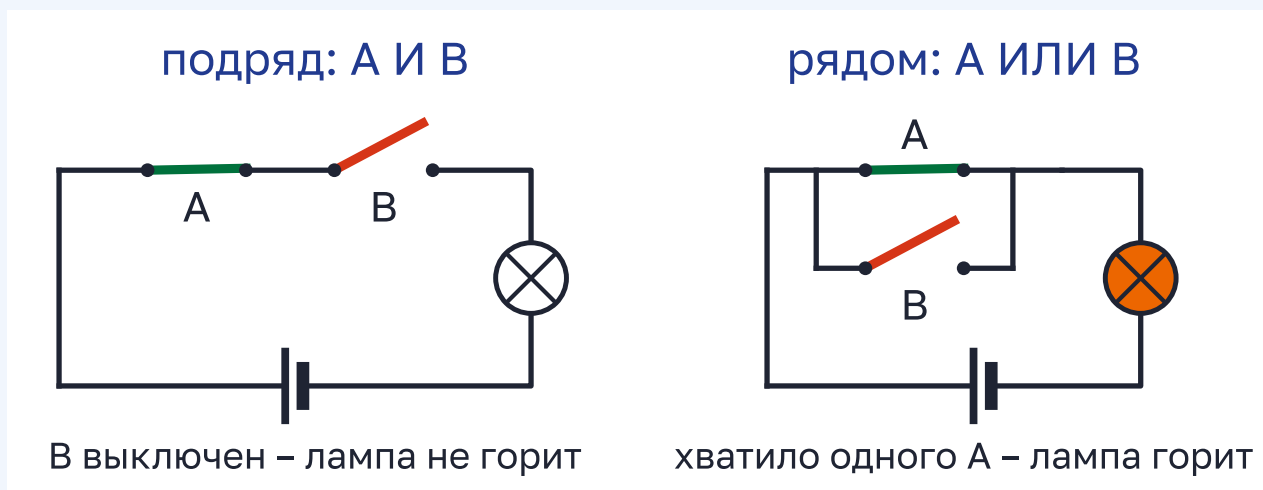


Рисунок 8.7 – Два выключателя и лампочка: подряд – это И, рядом – это ИЛИ

Выключатели подряд: ток пройдёт, только если замкнуты оба. Выключатели рядом: хватит любого одного.

Высказывание – фраза, про которую можно сказать «верно» или «неверно»: «в комнате темно», «пришёл бит 1», «уровень выше 40». Верно – 1 (**истина**), неверно – 0 (**ложь**). Совсем как бит. Из простых высказываний собирают сложные с помощью трёх связей.

- **НЕ А** переворачивает значение: было 1 – станет 0, было 0 – станет 1. «НЕ темно» верно, когда светло.
- **А И В** верно, только когда верны оба. Турникет в бассейне пустит, если есть абонемент И он не просрочен.
- **А ИЛИ В** верно, когда верно хотя бы одно из двух, а значит, и когда верны оба. Дверь подъезда откроется от ключа ИЛИ от кнопки домофона.

Бытовое «или» иногда значит «одно из двух, но не оба»: «на третье чай или компот». Логическое ИЛИ так не делает: оно верно и тогда, когда верно и то, и другое.

Таблица истинности – список всех вариантов входов и результат для каждого. У двух высказываний вариантов 4, у трёх – 8, у n высказываний – 2^n : каждое бывает 0 или 1, как бит в главе 4. Чтобы ни одного варианта не потерять, ставь строки по порядку двоичного счёта: 00, 01, 10, 11 или 000, 001, ..., 111.

Таблица 8.1

A	B	$A \text{ И } B$	$A \text{ ИЛИ } B$	$\text{НЕ } A$
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Если связок в выражении несколько, порядок такой: скобки, потом НЕ, потом И, потом ИЛИ. Похоже на арифметику, где умножение раньше сложения: И – это «умножение» ($1 \cdot 1 = 1$, всё остальное 0), а ИЛИ – «сложение», у которого больше единицы не бывает. Выражение $A \text{ ИЛИ } B \text{ И } C$ читается как $A \text{ ИЛИ } (B \text{ И } C)$. При $A = 0, B = 1, C = 1$ сначала $B \text{ И } C = 1$, потом 0 ИЛИ 1 = 1.

Длинное выражение считай по кусочкам, для каждого кусочка заведи свой столбик в таблице: сначала НЕ B , потом $A \text{ И НЕ } B$, потом всё целиком. Так ошибку видно сразу, в том столбике, где она случилась.

Логика встретится тебе в двух местах. В Кибериаде IDE на стрелке перехода пишут условие (глава 6) – это высказывание, которое машина проверяет. А в электронной таблице есть функции И, ИЛИ, НЕ: формула =ЕСЛИ(И(B2>40;C2=1);1;0) поставит 1, только если B2 больше 40 и в C2 стоит единица.

Запомни. И – нужны оба, ИЛИ – хватит одного (и оба тоже годятся), НЕ – наоборот. У n входов таблица истинности из 2^n строк, пиши их двоичным счётом. Порядок: скобки, НЕ, И, ИЛИ.

8.6. Комбинаторика: сколько вариантов кода

НА ПАЛЬЦАХ

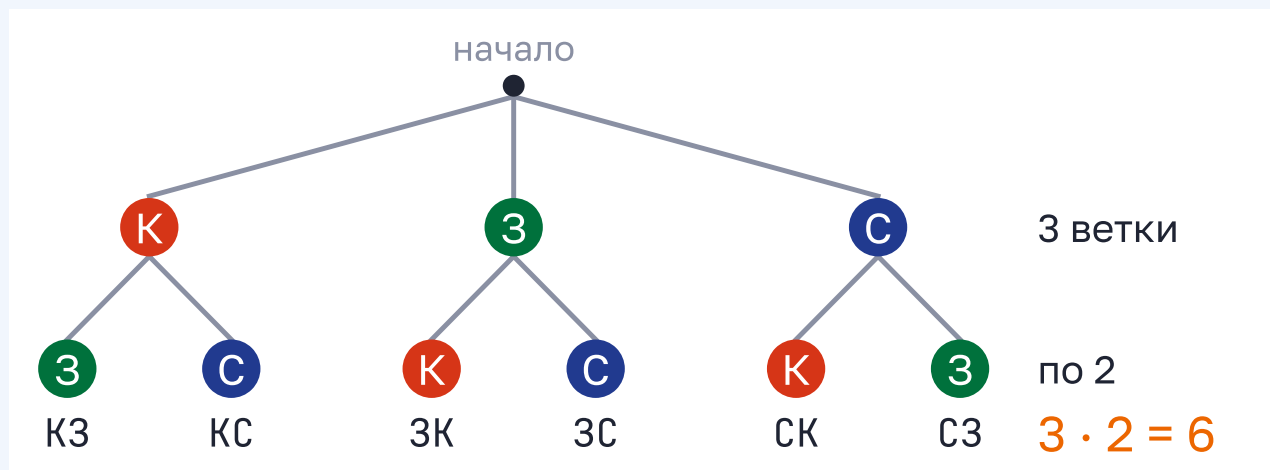


Рисунок 8.8 – Дерево вариантов: первая вспышка – любого из трёх цветов, вторая – любого, кроме того же самого

От каждой из трёх первых веток отходят по две. Считать листочки не нужно:
 $3 \cdot 2 = 6$.

Комбинаторика учит считать варианты, не выписывая их все. На ней держится вопрос «хватит ли кодов»: сколько команд поместится в три вспышки, сколько паролей из четырёх цифр, сколько букв можно закодировать точками и тире.

Правило умножения. Если выбор идёт по шагам, на первом шаге a вариантов, а на втором при каждом из них – по b вариантов, то всего вариантов $a \cdot b$. Шагов больше – множителей больше. На рисунке первая вспышка выбирается из трёх цветов, вторая из двух, и кодов выходит $3 \cdot 2 = 6$.

- **Знаки могут повторяться.** В коде k знаков, каждый выбирается из n : всего $n \cdot n \cdot \dots \cdot n = n^k$ кодов. Двоичный код из 8 бит – $2^8 = 256$ (глава 4). Фонарик трёх цветов и три вспышки – $3^3 = 27$.
- **Знаки не повторяются.** Из 5 разных команд робота собираем цепочку из 3 разных: первая – любая из 5, вторая – из оставшихся 4, третья – из 3. Всего $5 \cdot 4 \cdot 3 = 60$.
- **Всё переставить.** Три датчика подключают в три гнезда: в первое – любой из трёх, во второе – один из двух оставшихся, в третье – последний. $3 \cdot 2 \cdot 1 = 6$ способов. Такие варианты называют **перестановками**.
- **Есть запрет.** Трёхзначный номер не начинается с нуля: первая цифра – одна из 9, остальные – из 10. $9 \cdot 10 \cdot 10 = 900$ номеров, от 100 до 999.

Правило сложения. Если варианты делятся на группы, которые не пересекаются, считают каждую группу и складывают. Коды длиной 1, 2 или 3 бита: $2 + 4 + 8 = 14$ штук. Умножаем, когда выбираем «и то, и это» по шагам, складываем – когда берём «или то, или это».

Маленький случай всегда можно проверить перебором – выписать всё. Удобнее поручить это Python (подойдёт любой онлайн-интерпретатор):

```

from itertools import product          # берём готовый «перебиратель»
kody = list(product("K3C", repeat=2)) # все пары вспышек из цветов K, 3, C
print(len(kody))                      # 9 – сколько всего пар
print(kody[:4])                      # первые четыре пары

```

Первая строка подключает функцию `product` – она выдаёт все сочетания знаков. `repeat=2` значит «по два знака». `len` считает, сколько получилось, а `kody[:4]` показывает первые четыре: ('K', 'K'), ('K', '3'), ('K', 'C'), ('3', 'K'). Вышло $3 \cdot 3 = 9$, как и говорит правило умножения.

Запомни. По шагам – умножай, по отдельным группам – складывай. Повторы разрешены – n^k . Повторы запрещены – каждый следующий множитель на единицу меньше. Запрет на первом месте уменьшает только первый множитель.

8.7. Электронные таблицы

НА ПАЛЬЦАХ

	A	B	C
1	байт	бит в байте	бит
2	5	8	=A2*B2
3	12	8	=A3*B3
4	20	8	=A4*B4

↓
ПОТЯНУЛИ
ВНИЗ

2 → 3 → 4:
номер строки
в адресах едет

Рисунок 8.9 – Формулу в клетке C2 потянули вниз – адреса поехали вместе с ней

Таблица запоминает не «клетку A2», а «клетку на две левее в той же строке». Поэтому в каждой строке формула берёт свои числа.

Электронная таблица – тетрадь в клетку, которая сама считает. Подойдёт Excel или бесплатный LibreOffice Calc. Разработчики сферы говорят прямо: сигнал – это прежде всего ряд чисел. Таблица помогает этот ряд посчитать, отсеять и нарисовать графиком.

Столбцы обозначены буквами A, B, C..., строки – числами 1, 2, 3... Клетку (её ещё называют **ячейкой**) называют по столбцу и строке: B3 – столбец B, строка 3. В клетку можно вписать число, слово или **формулу**. Формула всегда начинается со знака «=»: =A2*B2 (звёздочка – умножение, косая черта – деление). После Enter в клетке видно результат, а саму формулу – в строке формул над таблицей.

Протягивание. Выдели клетку с формулой и потяни за маленький квадратик в её правом нижнем углу вниз. Формула скопируется в каждую клетку, и адреса в ней сдвинутся: =A2*B2 в строке 3 станет =A3*B3, в строке 4 – =A4*B4. Такие адреса называют **относительными**. Потянешь вправо – сдвинутся буквы: =A2*B2 превратится в =B2*C2.

Закреплённый адрес. Иногда одно число нужно всем строкам: порог, скорость канала, частота. Его кладут в отдельную клетку, например F1, а в формуле пишут \$F\$1. Знак доллара «прибивает» адрес гвоздём: при протягивании \$F\$1 остаётся \$F\$1. Такой адрес называют **абсолютным**. В Excel доллары расставляет клавиша F4, в LibreOffice Calc – Shift+F4. Бывает и половинка: в \$F1 закреплён только столбец, в F\$1 – только строка.

Пустая клетка в формуле считается нулём. Поэтому забытый доллар часто не выдаёт никакой ошибки: формула просто тихо сравнивает с нулём и выдаёт неверные числа.

Функции, без которых в задачах про сигнал не обойтись:

- =СУММ(B2:B9) – сумма клеток от B2 до B9 (двоеточие значит «от – до»);
- =СРЗНАЧ(B2:B9), =МАКС(B2:B9), =МИН(B2:B9) – среднее, наибольшее, наименьшее;
- =СЧЁТЕСЛИ(B2:B9;">40") – сколько чисел больше 40;
- =ЕСЛИ(B2>\$F\$1;1;0) – 1, если B2 выше порога из F1, иначе 0. Это детектор по уровню из главы 2, собранный в одной клетке.

Аргументы в русских версиях разделяют точкой с запятой, дробную часть пишут через запятую: =ОКРУГЛ(С2;1). Полный список формул книги – в справочнике.

Запомни. Формула начинается с «=». При протягивании обычные адреса едут, а адреса с долларами стоят на месте. Пустая клетка – это ноль. Общий для всех строк порог держи в отдельной клетке и ссылайся на неё через \$.

Совет. Прежде чем тянуть формулу вниз, спроси себя про каждый адрес: «он должен ехать вместе со строкой или стоять?» Потом щёлкни по клетке в середине столбца и посмотри формулу в строке формул – сразу видно, туда ли она смотрит.

8.8. Разбираем задачи

Задача 8.1 Сколько колебаний в одном бите

разминка · формула, перевод кГц и мс, период, скорость передачи

Передатчик отправляет биты звуком: бит 1 – тон частотой 2 кГц, бит 0 – тишина. Каждый бит длится 50 мс. а) Сколько колебаний тона укладывается в один бит? б) Какова скорость передачи в бит/с? в) Сколько миллисекунд длится одно колебание тона? Ответы введи через пробел.

Дано: $f = 2$ кГц; $\tau = 50$ мс.
Найти: N – ?; ν – ?; T – ? (в мс)

Решение

Шаг 1. Переводим единицы. Формулы записаны в герцах и секундах. $f = 2 \cdot 1000 = 2000$ Гц, $\tau = 50 : 1000 = 0,05$ с.

Шаг 2. Число колебаний. $N = f \cdot \tau = 2000 \cdot 0,05 = 100$ колебаний.

Шаг 3. Скорость передачи. Один бит длится 0,05 с. Сколько таких кусочков помещается в секунду? $\nu = 1 : \tau = 1 : 0,05 = 20$ бит/с.

Шаг 4. Период тона. $T = 1 : f = 1 : 2000 = 0,0005$ с. В миллисекунды: $0,0005 \cdot 1000 = 0,5$ мс.

Зайдём с другой стороны: в 50 мс по 0,5 мс на колебание помещается $50 : 0,5 = 100$ колебаний, столько же, сколько в шаге 2. А короткий путь из раздела 8.2 даёт то же без переводов: $50 \text{ мс} \cdot 2 \text{ кГц} = 100$.

 **Типичная ошибка.** Подставить числа как есть: $2 \cdot 50 = 100$ – случайно верно, но $1 : 50 = 0,02$ «бит/с» – уже чепуха, скорость вышла в тысячу раз меньше. Совпадение в шаге 2 случилось только потому, что «кило» и «милли» гасят друг друга. Надёжнее каждый раз переводить в герцы и секунды.

Ответ: 100 20 0,5.

Своя задача по программе сферы, модуль 4

Реши сам

8.1. ИК-передатчик мигает с частотой 38 кГц, один бит длится 0,5 мс. Сколько вспышек укладывается в один бит? Какова скорость передачи в бит/с? Ответы введи через пробел.

8.2. Скорость звукового канала – 25 бит/с, и в каждом бите 1 успевает пройти 60 колебаний тона. Какова частота тона в герцах?

Задача 8.2 Три завода

деление в отношении, процент от числа, доля в процентах

Для кружка закупили 400 ИК-приёмников с трёх заводов. С первого завода взяли в 3 раза больше приёмников, чем со второго, и в 2 раза меньше, чем с третьего. Бракованных у первого завода 5 %, у второго – 10 %, у третьего – 2,5 %. Сколько всего бракованных приёмников в закупке и какой это процент от всех? Ответы введи через пробел.

Дано: всего 400 шт.; первый = 3 · второй; третий = 2 · первый; брак 5 %, 10 %, 2,5 %.

Найти: число бракованных; их доля, %.

Решение

Шаг 1. Части. Удобнее всего мерить всё приёмниками второго завода. Пусть их одна часть. У первого – в 3 раза больше, 3 части. У третьего – вдвое больше, чем у первого, $2 \cdot 3 = 6$ частей. Всего $1 + 3 + 6 = 10$ частей.

Шаг 2. Сколько штук с каждого завода. Одна часть – $400 : 10 = 40$ приёмников. Первый завод: $3 \cdot 40 = 120$, второй – 40, третий – $6 \cdot 40 = 240$. Проверка: $120 + 40 + 240 = 400$.

Шаг 3. Брак по заводам.

- первый: $120 \cdot 5 : 100 = 6$ шт.;
- второй: $40 \cdot 10 : 100 = 4$ шт.;
- третий: $240 \cdot 2,5 : 100 = 6$ шт.

Шаг 4. Всего и в процентах. Бракованных $6 + 4 + 6 = 16$ шт. Доля: $16 : 400 \cdot 100 = 4$ %.

Больше всего приёмников пришло с третьего завода, где брака меньше всех. Поэтому общий процент и получился ближе к его 2,5 %, чем к 10 % второго завода.

⚠ Типичная ошибка. Сложить три процента и поделить на три: $(5 + 10 + 2,5) : 3 \approx 5,8$ %. Так можно делать, только если с каждого завода взяли поровну. Здесь со второго, самого бракованного, взяли меньше всех, и его 10 % весят мало. Проценты от разных количеств не усредняют – считают штуки, потом делят на общее число.

Ответ: 16 4.

По мотивам задачи первого отборочного этапа НТО 2025/26, профиль «Интеллектуальные энергетические системы», «Условная надёжность»

Реши сам

8.3. В кабинете три партии датчиков звука: 150 штук с браком 4 %, 250 штук с браком 2 % и 100 штук с браком 6 %. Какой процент брака во всех датчиках вместе?

8.4. По каналу отправили 2000 пакетов, 4 % из них потерялись. После ремонта антенны отправили ещё 2000, и потерялось 1,5 %. а) Сколько пакетов потерялось в первый и во второй раз? б) На сколько процентов уменьшилось число потерянных пакетов? Ответы введи через пробел.

Задача 8.3 Сколько байт в пакете

уравнение из формулы, обратные действия, байты и биты

Датчик отправляет пакет по ИК-каналу со скоростью 80 бит/с. В пакете сначала идут 16 служебных бит (адрес и проверка), потом сообщение – n байт. Приёмник засёк, что пакет шёл 2,2 с. Сколько байт было в сообщении?

Дано: $v = 80$ бит/с; служебных 16 бит; $t = 2,2$ с.

Найти: n – ? (байт)

Решение

Шаг 1. Формула времени. Весь пакет – это $16 + 8n$ бит (байт умножаем на 8). Время = объём : скорость:

$$\frac{16 + 8n}{80} = 2,2.$$

Шаг 2. Убираем деление. Умножаем обе части на 80: $16 + 8n = 2,2 \cdot 80 = 176$. По смыслу это объём всего пакета: 176 бит.

Шаг 3. Убираем служебные биты. Вычитаем 16 из обеих частей: $8n = 160$. На сообщение пришлось 160 бит.

Шаг 4. Биты в байты. Делим на 8: $n = 20$ байт.

Шаг 5. Подстановка. $(16 + 8 \cdot 20) : 80 = 176 : 80 = 2,2$ с – как у приёмника.

Двадцать байт – это, например, двадцать букв латиницей. Для короткого сообщения датчика размер правдоподобный.

⚠ Типичная ошибка. Забыть служебные биты и сразу поделить: $176 : 8 = 22$ байта. Или вычесть 16 из 2,2 – складывать и вычитать можно только величины одного рода, а здесь биты смешали с секундами. Сначала переведи время в биты, потом убирай служебные.

Ответ: 20.

Своя задача по программе сферы, модуль 4

Реши сам

8.5. Пакет идёт со скоростью 50 бит/с, в нём 10 служебных бит и n байт сообщения. Передача заняла 4,2 с. Найди n .

8.6. Сообщение передают короткими и длинными тонами, паузы не считаем. Сообщение из 3 коротких и 2 длинных тонов заняло 0,9 с, а из 3 коротких и 5 длинных – 1,8 с. Сколько миллисекунд длится короткий тон и сколько – длинный? Ответы введи через пробел.

Задача 8.4 Когда завоет сирена

логика, И, ИЛИ, НЕ, таблица истинности

На даче стоит самодельная сигнализация. У неё три датчика-высказывания:

- A – «дверь открыта»;
- B – «хозяин ввёл код»;
- C – «датчик дыма сработал».

Сирена включается по правилу $S = (A \text{ И НЕ } B) \text{ ИЛИ } C$. Составь таблицу истинности, строки расставь двоичным счётом от $ABC = 000$ до 111 . В ответ введи столбец S сверху вниз – восемь цифр подряд.

Дано: $S = (A \text{ И НЕ } B) \text{ ИЛИ } C$; три входа.

Найти: столбец S для восьми строк.

Решение

Шаг 1. Сколько строк. Входов три, у каждого два значения: $2^3 = 8$ строк. Выписываем их двоичным счётом.

Шаг 2. Разбиваем правило на кусочки. Сначала скобки, внутри – сначала НЕ. Для скобки заведём отдельный столбец $A \text{ И НЕ } B$, а потом посчитаем всё правило.

Шаг 3. Заполняем.

Таблица 8.2

<i>A</i>	<i>B</i>	<i>C</i>	<i>A И НЕ B</i>	<i>S</i>
0	0	0	0	0
0	0	1	0	1
0	1	0	0	0
0	1	1	0	1
1	0	0	1	1
1	0	1	1	1
1	1	0	0	0
1	1	1	0	1

НЕ *B* – это *B* наоборот: единица там, где *B* = 0. Значит, *A И НЕ B* равно 1, только когда в строке *A* = 1 и *B* = 0: строки 5 и 6. В последнем столбце ставим 1, если единица есть хотя бы в одном из двух: в столбце «*A И НЕ B*» или в столбце *C*.

Шаг 4. Читаем по смыслу. Сирена воет при любом дыме (все строки с *C* = 1) и ещё в строке 5: дверь открыта, кода нет, дыма нет – залез чужой. Хозяин, который открыл дверь и ввёл код, сирену не включает (строка 7). Для дачи правило годится.

Если есть компьютер, таблицу можно получить и программой:

```
from itertools import product
for A, B, C in product([0, 1], repeat=3):
    S = (A and not B) or C
    print(A, B, C, int(S))
```

восемь строк: 000, 001, ... 111
and – И, or – ИЛИ, not – НЕ
int делает из «правды» 1, из «лжи» 0

⚠ Типичная ошибка. Забыть про НЕ и считать (*A И B*) ИЛИ *C*: тогда сирена завоет на хозяина с кодом и промолчит, когда в дом залезет чужой. Вторая ошибка – путать порядок строк: если в таблице строки идут вразнобой, столбец ответа тоже перемешается, и компьютер его не примет.

Ответ: 01011101.

Своя задача по программе сферы, модуль 6

Реши сам

8.7. КиберМишка мигает по правилу $Z = (A \text{ ИЛИ } B) \text{ И НЕ } C$, где *A* – «громкий звук», *B* – «хлопок в ладоши», *C* – «включён ночной режим». Найди *Z* для четырёх случаев (*A*, *B*, *C*): (1, 0, 0), (0, 0, 1), (1, 1, 1), (0, 1, 0). В ответ введи четыре цифры подряд.

8.8.* В главе 5 каждый бит передавали трижды, а приёмник голосовал большинством. Голосование можно записать логикой: $M = (A \text{ и } B) \text{ или } (A \text{ и } C) \text{ или } (B \text{ и } C)$, где A, B, C – три принятые копии бита. а) Составь таблицу истинности для M (строки от 000 до 111) и запиши столбец M . б) Приёмник получил тройки 110, 001, 101, 010. Какие четыре бита он выдаст? Ответы введи через пробел.

Задача 8.5 Сколько сигналов у фонарика

правило умножения, запрет на повтор, правило сложения

Фонарик для передачи команд светит тремя цветами: красным (К), зелёным (З) и синим (С). Команда – это несколько вспышек подряд без пауз. а) Сколько разных команд из 3 вспышек, если цвета могут повторяться? б) Приёмник не отличает две одинаковые вспышки подряд от одной длинной, поэтому соседние вспышки должны быть разного цвета. Сколько теперь команд из 3 вспышек? в) Сколько команд длиной от 1 до 3 вспышек можно составить с тем же правилом? Ответы введи через пробел.

Дано: 3 цвета; 3 вспышки; соседние разные (в пунктах б и в).

Найти: число команд в пунктах а, б, в.

Решение

Шаг 1. Повторы можно. Первая вспышка – любой из 3 цветов, вторая – тоже любой из 3, третья – тоже. $3 \cdot 3 \cdot 3 = 27$ команд.

Шаг 2. Соседние разные. Первая – любой из 3. Вторая – любой, кроме цвета первой: 2 варианта. Третья – любой, кроме цвета второй: снова 2. Цвет первой вспышки третьей не мешает: КЗК – законная команда. $3 \cdot 2 \cdot 2 = 12$ команд.

Шаг 3. Разные длины. Команды из одной, двух и трёх вспышек – три отдельные группы, поэтому считаем каждую и складываем. Одна вспышка: 3. Две: $3 \cdot 2 = 6$. Три: 12 из шага 2. Всего $3 + 6 + 12 = 21$.

Сколько это на практике? Двадцать одна команда – с запасом хватит, чтобы управлять игрушечной машинкой: вперёд, назад, повороты, скорость, фары.

Перебор на Python выписывает все команды и считает те, что подходят:

```
from itertools import product
def mozhno(kod):
    return all(kod[i] != kod[i + 1] for i in range(len(kod) - 1))
print(len(list(product("K3C", repeat=3))))
print(sum(mozhno(k) for k in product("K3C", repeat=3)))
print(sum(mozhno(k) for n in (1, 2, 3) for k in product("K3C", repeat=n)))
```

Функция `mozhno` проверяет каждую пару соседей: `all` отвечает «да», только если все пары разные. `sum` считает, сколько раз вышло «да».

❗ **Типичная ошибка.** В пункте б считать третью вспышку из одного цвета: $3 \cdot 2 \cdot 1 = 6$. Это ответ на другой вопрос – «все три цвета разные». По условию третья вспышка должна отличаться только от второй. В пункте в часто перемножают группы, $3 \cdot 6 \cdot 12$, хотя команда бывает или короткой, или длинной – группы складывают.

Ответ: 27 12 21.

Своя задача по программе сферы, модуль 4

Реши сам

8.9. У самодельного пульта 5 кнопок. Команда – это две кнопки, нажатые друг за другом, можно одну и ту же дважды. Сколько разных команд можно задать?

8.10. В азбуке Морзе каждая буква – цепочка из точек и тире. Сколько разных цепочек длиной от 1 до 4 знаков можно составить? Скольким буквам русского алфавита (в нём 33 буквы) не хватило бы таких коротких кодов? Ответы введи через пробел.

Задача 8.6 Порог в отдельной клетке

электронные таблицы, ЕСЛИ, закреплённый адрес, СУММ

Шестиклассница Вера записала сигнал с датчика в таблицу (рисунок ниже): в столбце А – время в миллисекундах, в столбце В – уровень сигнала. Порог лежит в клетке F1. В C2 Вера написала `=ЕСЛИ(В2>$F$1;1;0)` и протянула формулу до C9, а в C10 написала `=СУММ(С2:С9)`. а) Что окажется в C5? б) Что покажет C10? в) Что покажет C10, если в F1 вписать 50? Ответы введи через пробел.

C2 <code>=ЕСЛИ(В2>\$F\$1;1;0)</code>						
	А	В	С	Д	Е	F
1	t, мс	уровень	выше?		порог	40
2	0	12	?			
3	10	45	?			
4	20	63	?			
5	30	38	?			
6	40	71	?			
7	50	55	?			
8	60	20	?			
9	70	48	?			
10	всего		?			

Рисунок 8.10 – Таблица сигнала. В столбце С – формула из условия, её значения пока скрыты

Дано: B2:B9 = 12, 45, 63, 38, 71, 55, 20, 48; F1 = 40; формула `=ЕСЛИ(B2>$F$1;1;0)`, протянута до С9.

Найти: С5; С10 при F1 = 40; С10 при F1 = 50.

Решение

Шаг 1. Что делает формула. «Если уровень в этой строке больше порога из F1 – поставь 1, иначе 0». При протягивании B2 едет вниз (B3, B4...), а `$F$1` стоит на месте: все строки сравниваются с одним и тем же порогом.

Шаг 2. Клетка С5. В С5 формула стала `=ЕСЛИ(B5>$F$1;1;0)`. В B5 стоит 38, а $38 > 40$ неверно. В С5 будет 0.

Шаг 3. Столбец С при пороге 40. Идём по строкам: 12 – нет (0), 45 – да (1), 63 – да (1), 38 – нет (0), 71 – да (1), 55 – да (1), 20 – нет (0), 48 – да (1). Столбец: 0, 1, 1, 0, 1, 1, 0, 1.

Шаг 4. Сумма. $0 + 1 + 1 + 0 + 1 + 1 + 0 + 1 = 5$. Сумма столбца из нулей и единиц – это количество единиц: сколько раз сигнал был выше порога.

Шаг 5. Порог 50. Формулы трогать не нужно, таблица пересчитает всё сама. Больше 50 только 63, 71 и 55. С10 покажет 3. Для этого порог и держат в отдельной клетке: поменял одно число – пересчитался весь столбец.

Для самопроверки подойдёт функция из раздела 8.7: `=СЧЁТЕСЛИ(B2:B9;">40")` тоже даёт 5, а с `">50"` – 3.

 **Типичная ошибка.** Написать в С2 формулу без долларов, `=ЕСЛИ(B2>F1;1;0)`. При протягивании в С3 окажется `=ЕСЛИ(B3>F2;1;0)`, в С4 – `B4>F3`, и так далее. Клетки F2–F8 пустые, то есть нули, а любой уровень больше нуля. Таблица не ругается – она молча ставит единицы везде, кроме С2, и сумма выходит 7 вместо 5.

Ответ: 0 5 3.

Своя задача по программе сферы, модуль 2

Реши сам

8.11. В A2 записана частота тона 2500 Гц, в A3 – 500 Гц. В B2 написали `=1000/A2` – так таблица считает период тона в миллисекундах – и протянули формулу на B3. Сначала выпиши для себя, какая формула окажется в B3. Какие числа покажут B2 и B3? В ответ введи эти два числа через пробел.

8.12. В E1 записана скорость канала 250 бит/с. В B2, B3, B4 – размеры сообщений в байтах: 50, 125, 400. В C2 написали `=B2*8/$E$1` и протянули до C4. Какие числа покажут C2, C3, C4? Что покажет C3, если в формуле забыть доллары? Ответы введи через пробел; если в клетке появится ошибка деления на ноль, напиши вместо числа «ошибка».

Задача 8.7* Где поставить пульт

система уравнений, вычитание уравнений, расстояния на прямой

Это самая трудная задача главы: её идея пришла из олимпиады для 8–9 классов. Незвестных тут три, но новых приёмов не нужно – то же вычитание уравнений, что с короткими и длинными тонами в разделе 8.4, только два раза подряд.

Вдоль прямой дороги к Обскому морю стоят четыре ретранслятора, по порядку: 1, 2, 3 и 4. Расстояния между ними неизвестны. Инженеры прикинули, сколько кабеля нужно, чтобы протянуть от каждого ретранслятора свою линию к пульта управления, если пульт поставить рядом с одним из них. Пульт у ретранслятора 1 – всего 31 км кабеля. Пульт у ретранслятора 3 – 19 км. Пульт у ретранслятора 4 – 29 км. Каково расстояние между ретрансляторами 1 и 3 в километрах?

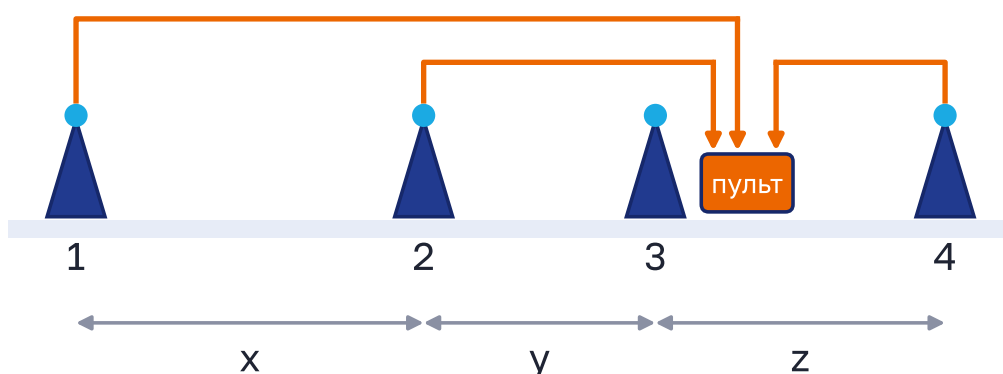


Рисунок 8.11 – Четыре ретранслятора на дороге, расстояния x , y , z неизвестны. Если пульт стоит у ретранслятора 3, кабели идут от 1, 2 и 4

Дано: пульт у 1 – 31 км; у 3 – 19 км; у 4 – 29 км.

Найти: расстояние от 1 до 3, км.

Решение

Шаг 1. Обозначения. Пусть x – от 1 до 2, y – от 2 до 3, z – от 3 до 4 (в километрах). Нам нужно $x + y$.

Шаг 2. Кабель при каждом положении пульта. Складываем длины линий от остальных трёх ретрансляторов.

- Пульт у 1: от 2 – x , от 3 – $x + y$, от 4 – $x + y + z$. Всего $3x + 2y + z = 31$.
- Пульт у 3: от 1 – $x + y$, от 2 – y , от 4 – z . Всего $x + 2y + z = 19$.
- Пульт у 4: от 1 – $x + y + z$, от 2 – $y + z$, от 3 – z . Всего $x + 2y + 3z = 29$.

Откуда, например, $3x + 2y + z$? Посчитай, в скольких линиях встречается каждый отрезок: x – в трёх, y – в двух, z – в одной.

Шаг 3. Первое минус второе. $2y$ и z есть в обоих, при вычитании они исчезнут: $(3x + 2y + z) - (x + 2y + z) = 31 - 19$, то есть $2x = 12$, $x = 6$ км. По смыслу: пульт переехал от 1 к 3, и кабель укоротился на двойное x .

Шаг 4. Третье минус второе. $(x + 2y + 3z) - (x + 2y + z) = 29 - 19$, $2z = 10$, $z = 5$ км. Здесь пульт переехал от 4 к 3, и кабель укоротился на двойное z .

Шаг 5. Находим y . Подставляем во второе уравнение: $6 + 2y + 5 = 19$, $2y = 8$, $y = 4$ км.

Шаг 6. Ответ и проверка. От 1 до 3: $x + y = 6 + 4 = 10$ км. Подставим всё в первое и третье: $3 \cdot 6 + 2 \cdot 4 + 5 = 31$, $6 + 2 \cdot 4 + 3 \cdot 5 = 29$ – оба равенства верны.

Самый короткий кабель вышел, когда пульт стоит у ретранслятора 3, ближе к середине цепочки. Так и бывает: из середины до всех ближе, чем с края.

 **Типичная ошибка.** Посчитать кабель от ретранслятора 1 как $x + y + z$ – только до самого дальнего. Линия нужна от каждого ретранслятора своя, поэтому отрезок x входит в сумму трижды. Помогает рисунок: проведи пальцем по каждой линии и отмечай, какие отрезки она проходит.

Ответ: 10.

По мотивам задачи первого отборочного этапа НТО 2025/26, профиль «Технологии беспроводной связи», предметный тур по математике (задача про дома на улице Лобачевского, 8–9 классы)

Реши сам

8.13. Три вышки стоят на прямой в порядке А, Б, В. Если пульт поставить у вышки А, на линии от Б и В уйдёт 26 км кабеля. Если у вышки В – на линии от А и Б уйдёт 22 км. Каково расстояние от А до В?

8.14.* Три пакета данных отправили парами. Первый и второй вместе – 130 байт, второй и третий – 170 байт, первый и третий – 160 байт. Сколько байт в каждом пакете? Ответы введи через пробел по порядку.

Ответы и решения

Сначала реши задачу сам и только потом сверяйся. Не сошлось – найди в решении шаг, на котором твой путь разошёлся с нашим.

К главе 1. Что такое киберфизическая система

1.1. Ответ: 5123.

1.2. Ответ: а) 15; б) 23.

Подсказка: модуль Wi-Fi ничего не измеряет и ничего не двигает – это передача.

1.3. Ответ: 24.

Подсказка: у обычного выключателя решает человек, у игры нет настоящего мира, у часов нет ни датчика, ни программы.

1.4. Ответ: 246.

1.5. Ответ: 24351.

1.6. Ответ: 2351 (лишнее – событие 4).

1.7. Ответ: 10 м.

1.8. Ответ: 7.

Подсказка: задача обратная – найди, сколько времени есть на всю реакцию ($0,2 : 0,5$ м/с), отними известные задержки, а число опросов в секунду округли вверх.

1.9. Ответ: 2,56 с.

1.10. Ответ: 396 млн км.

Подсказка: сначала отними 6 минут раздумий, остаток подели пополам – это путь сигнала в одну сторону.

1.11. Ответ: -3.

1.12. Ответ: 11.

1.13. Ответ: 105 мин.

1.14. Ответ: 30 мин.

Подсказка: здесь «включить» стоит у верхнего порога, а «выключить» – у нижнего.

К главе 2. Сигнал и его параметры

2.1. Ответ: 20 мс.

2.2. Ответ: 0,5 – половина периода; звуки гасят друг друга, и человек почти ничего не услышит.

Подсказка: сначала найди период: $1 : 250 \text{ с} = 4 \text{ мс}$.

2.3. Ответ: 250 Гц.

2.4. Ответ: амплитуда 60 ед., частота 2 Гц.

2.5. Ответ: 7 938 000 отсчётов.

2.6. Ответ: 5000 Гц.

Подсказка: отсчётов на период – это $f_{\text{д}} : f$; здесь это отношение должно быть не меньше 10.

2.7. Ответ: 3.

2.8. Ответ: 4 2.

Подсказка: скачки 65 и 70 делятся по одному отсчёту – по новому правилу это шум, а не хлопок.

2.9. Ответ: 2 м/с.

2.10. Ответ: 3000 оборотов в минуту.

Подсказка: один оборот – это четыре щелчка, то есть 20 мс.

2.11. Ответ: в 3,2 раза.

2.12. Ответ: 200 68.

Подсказка: гребень из 1,0 мс у второго микрофона – в 3,0 мс. Вариант «на период позже», в 8,0 мс, проверь условием про 1 м.

К главе 3. Разложение сигнала: спектр и фильтры

3.1. Ответ: 245.

3.2. Ответ: 3 0,5 нет.

Подсказка: выше суммы амплитуд $2 + 1 + 0,5 = 3,5$ сигнал не поднимется.

3.3. Ответ: 123.

3.4. Ответ: 23.

Подсказка: второй фильтр работает с тем, что осталось после первого: из волн 1, 2, 3 он убирает всё, что ниже 300 Гц.

3.5. Ответ: 18 20 19 21.

3.6. Ответ: 15 16.

Подсказка: сумма первого окна $14 \cdot 3 = 42$, а известные два числа дают $12 + 15 = 27$.

3.7. Ответ: 7.

Подсказка: при окне 3 окно целиком помещается внутри вспышки, и её значение остаётся 6, но блик $4:3 \approx 1,33$.

3.8. Ответ: 6.

Подсказка: сумма вспышки 24, нужно $24:k$ не меньше 4.

3.9. Ответ: 50.

Подсказка: здесь даны две частоты, поэтому делим одну на другую.

3.10. Ответ: 2,5 0,5 10.

3.11. Ответ: 110.

Подсказка: остальные частоты кратны 25 Гц, а $110:25 = 4,4$ – не целое.

3.12. Ответ: 14.

Подсказка: сначала найди кусок, который повторяется (2 7 7 4 9), потом сверь с ним каждую пятёрку.

3.13. Ответ: 20.

Подсказка: на периоды наводок приходится 4 и 5 измерений; окно должно вмещать целое число и тех и других – ищи наименьшее число, которое делится и на 4, и на 5.

3.14. Ответ: 1245.

Подсказка: окно 12 длится $12:120 = 0,1$ с; за это время должно пройти целое число периодов, а у 25 Гц выходит $25 \cdot 0,1 = 2,5$.

К главе 4. Кодирование и системы счисления

4.1. Ответ: 101.

4.2. Ответ: Лёша записал 154, Юля прочитала 89, разница 65.

Подсказка: справа налево байт читается как 01011001.

4.3. Ответ: 01111111, это 127.

4.4. Ответ: MISHA.

Подсказка: отбрось первое показание; каждая стрелка – одна шестнадцатеричная цифра, например 4 и 13 – это $4 \cdot 16 + 13 = 77$, буква М.

4.5. Ответ: 128 знаков.

4.6. Ответ: а) 6 бит; б) 108 байт.

Подсказка: разных значений 61 – ноль тоже считается; за сутки 144 измерения.

4.7. Ответ: 50 бит/с; 150 бит.

4.8. Ответ: 4 с.

Подсказка: 16 цветов – это 4 бита на точку, а точек $16 \cdot 16 = 256$.

4.9. Ответ: а) 11101101 и 01000001 ; б) 94.

4.10. Ответ: CODE.

Подсказка: нечётными получаются строка 3 и столбец 2; после исправления строка 3 – это 1101 , то есть D.

4.11. Ответ: 27 единиц; 2,7 с.

4.12. Ответ: ИДИ ДОМОЙ.

4.13. Ответ: АБВАГ.

4.14. Ответ: а) АВ или БА; б) код А заменить на 00 – получится 00 , 01 , 10 .

Подсказка: ломает код то слово, которое служит началом другого; его и нужно удлинять.

К главе 5. Модуляция и каналы связи

5.1. Ответ: 11010010.

5.2. Ответ: 12 111001000011.

Подсказка: при 20 бит/с один бит длится 0,05 с – каждую длительность подели на 0,05.

5.3. Ответ: 60 60.

Подсказка: сначала переведи команды в биты по таблице: 10 00 11, потом каждый бит – 20 мс.

5.4. Ответ: 30 10100110.

Подсказка: порог должен быть выше самой громкой «тишины» (22) и ниже самого тихого тона (55).

5.5. Ответ: 12 6.

5.6. Ответ: 2000 4000 1001101.

Подсказка: частота по периоду – $f = 1 : T$, а 0,25 мс = 0,00025 с.

5.7. Ответ: 257.

5.8. Ответ: 8.

Подсказка: перебирай длительности по порядку и проверяй оба условия; при 6 мс у частоты 1500 Гц только 9 колебаний.

5.9. Ответ: 3 с.

5.10. Ответ: 41.

Подсказка: с повтором одна команда занимает $4 \cdot 3 = 12$ бит, а стартовую метку вычти из всех бит до деления.

5.11. Ответ: 1 2.

5.12. Ответ: 10101011 5.

Подсказка: испорченная тройка – та, где цифры не все одинаковые.

5.13. Ответ: 110010 011001.

Подсказка: на 2500 Гц «тон» слабый, 12–14, поэтому порог для этого канала нужен свой, около 7.

5.14. Ответ: 3 2,1.

Подсказка: ИК-часть идёт $80 : 40 = 2$ с, звуковая – $40 : 20 = 2$ с плюс $34 : 340 = 0,1$ с в пути; ждать надо ту, что придёт позже.

К главе 6. ПРИМС: машины состояний в Кибериаде

6.1. Ответ: «Бегущий огонь».

Подсказка: второе долгое нажатие заставит гирлянду уже выключенной и пропадает.

6.2. Ответ: «Слабо»; ярко – 33 раза.

Подсказка: круг – 3 нажатия, $100 = 3 \cdot 33 + 1$; ярко загорается на 2-м, 5-м, 8-м... нажатии.

6.3. Ответ: 4 72 (пропали 4 нажатия, последний раз «А» наступило в 72 с).

6.4. Ответ: 3.

Подсказка: жми в первую же секунду, когда условие верно: раньше нажмёшь – раньше светофор вернётся в «А». Нажатия в 20, 62 и 104 с (круг 22 с плюс 20 с ожидания), зелёный пешеходам – в 23, 65 и 107 с.

6.5. Ответ: 25.

6.6. Ответ: при 20 °С и 21 °С; условия должны быть $[t < 15]$ и $[t > 28]$.

Подсказка: ищи числа, для которых верны оба условия сразу.

6.7. Ответ: стрелка 2; последний раз будильник зазвенел в 7:10.

6.8. Ответ: 4 стрелки без рамки, 2 – с рамкой.

6.9. Ответ: 7 46.

6.10. Ответ: никогда не наступит 3 – «Готово»; чайник останется в «Греет».

Подсказка: стрелка в «Готово» есть, но её условие никогда не бывает верным.

6.11. Ответ: «Громко»; в «Паузе» – 1 раз.

6.12. Ответ: 7.

Подсказка: «разряд» – из всех четырёх вложенных, «стоп» – только из трёх: у «Сигналю» своя стрелка.

6.13. Ответ: такой замок – на 6-м нажатии, правильный – на 3-м.

6.14. Ответ: в «1-1»; правильный замок откроется на 6-м нажатии, а замок со сбросом в «Пусто» не откроется совсем.

К главе 7. Управление КиберМишкой

7.1. Ответ: 2 3 Ю.

7.2. Ответ: 6.

Подсказка: финиш сзади и левее – три шага «Назад», поворот налево и два шага вперёд.

7.3. Ответ: 26 54.

7.4. Ответ: 10 74.

Подсказка: клеток в рамке квадрата со стороной $n - 4$ $n - 4$; реши $4n - 4 = 36$.

7.5. Ответ: 3 3.

7.6. Ответ: 2 1.

Подсказка: после второй тревоги счётчик равен 2, условие «счётчик < 2» ложно, и хлопки в «Слушаю» пропадают.

7.7. Ответ: 5.

Подсказка: отсчитывай 600 мс от последнего **засчитанного** хлопка, а не от последнего услышанного.

7.8. Ответ: 1000.

Подсказка: проверь все три порога; при низком пороге соседние всплески сливаются в один.

7.9. Ответ: 12103.

7.10. Ответ: 6703.

Подсказка: остаток от деления на 8 – три последние двоичные цифры.

7.11. Ответ: 3 1.

7.12. Ответ: 2 0.

7.13. Ответ: 2 3 3.

7.14. Ответ: 1 2.

Подсказка: теперь переворот в «Показываю» обрабатывает само «Показываю», до «Работы» он не доходит.

К главе 8. Математика киберфизика

8.1. Ответ: 19 2000.

8.2. Ответ: 1500 Гц.

Подсказка: сначала длительность бита $\tau = 1 : 25 = 0,04$ с, потом $f = N : \tau$ – треугольник из раздела 8.1.

8.3. Ответ: 3,4 %.

8.4. Ответ: 80 30 62,5.

Подсказка: сравнивай число потерянных пакетов с тем, что было до ремонта, – с 80; разница процентов 2,5 – это другой вопрос.

8.5. Ответ: 25 байт.

8.6. Ответ: 100 300.

Подсказка: коротких в обоих сообщениях поровну – вычти первое уравнение из второго, и останутся только длинные тоны.

8.7. Ответ: 1001.

8.8. Ответ: 00010111 1010.

Подсказка: $M = 1$ там, где в строке хотя бы две единицы; это и есть голосование большинством.

8.9. Ответ: 25 команд.

8.10. Ответ: 30 3.

Подсказка: цепочки разной длины – отдельные группы: $2 + 4 + 8 + 16$.

8.11. Ответ: 0,4 2.

Подсказка: в В3 окажется $=1000/A3$ – адрес поехал вместе со строкой.

8.12. Ответ: 1,6 4 12,8 ошибка.

Подсказка: без долларов в С3 окажется $=B3*8/E2$, а пустая Е2 – это ноль, на ноль делить нельзя.

8.13. Ответ: 16 км.

Подсказка: здесь уравнения не вычитают, а складывают: $3x + 3y = 48$, и сразу видно $x + y$.

8.14. Ответ: 60 70 100.

Подсказка: сложи все три суммы – каждый пакет войдёт дважды, $2 \cdot (\text{все три}) = 460$; дальше вычитай из 230 каждую пару.

Справочник

Всё, что может понадобиться под рукой во время решения: числа и единицы, формулы и правила из глав, азбука Морзе, мини-шпаргалки по Кибериаде IDE, таблицам и Python, словарь и все «Запомни» из глав на нескольких страницах.

Числа, которые используются в книге

Если в условии задачи на отборе дано другое значение, бери его из условия.

Таблица С.1

Величина	Значение
Скорость звука в воздухе	около 340 м/с
Скорость света (и радиоволн, и инфракрасного луча)	около 300 000 км/с
Какие звуки слышит человек	примерно от 20 Гц до 20 000 Гц
Нота «ля» первой октавы, по ней настраивают инструменты	440 Гц
Частота тока в розетке в России	50 Гц
FM-радиостанции	от 87,5 до 108 МГц
Частота дискретизации звука на музыкальном компакт-диске	44 100 Гц
Один байт	8 бит
Сколько значений помещается в один байт	256: от 0 до 255

Звук ниже 20 Гц называют инфразвуком, выше 20 000 Гц – ультразвуком. Человек их не слышит, а приборы ловят. Инфракрасный свет глаз тоже не видит. Зато его часто видит камера телефона: наведи её на пульт от телевизора и нажми кнопку – на экране замигает огонёк.

Перевод единиц

Таблица С.2

Что	Как переводить
Время	$1 \text{ с} = 1000 \text{ мс} \cdot 1 \text{ мс} = 1000 \text{ мкс} \cdot 1 \text{ мин} = 60 \text{ с}$
Частота	$1 \text{ кГц} = 1000 \text{ Гц} \cdot 1 \text{ МГц} = 1000 \text{ кГц} \cdot 1 \text{ ГГц} = 1000 \text{ МГц}$
Частота и период	$1 \text{ Гц} - \text{период } 1 \text{ с} \cdot 1 \text{ кГц} - 1 \text{ мс} \cdot 1 \text{ МГц} - 1 \text{ мкс}$
Длина	$1 \text{ км} = 1000 \text{ м} \cdot 1 \text{ м} = 100 \text{ см} = 1000 \text{ мм}$

Что	Как переводить
Информация	1 байт = 8 бит · 1 Кбайт = 1024 байта · 1 Мбайт = 1024 Кбайт
Скорость передачи	1 кбит/с = 1000 бит/с · 1 байт/с = 8 бит/с

Совет. Приставка «кило» в задачах про информацию бывает двух видов. Объём памяти в школьной информатике считают по 1024: 1 Кбайт = 1024 байта. Частоту и скорость связи – по 1000: 1 кГц = 1000 Гц, 1 кбит/с = 1000 бит/с. Если в условии сказано иначе, решай по условию.

Формулы и правила

Если в условии задачи правило записано по-своему, решай по условию.

Сигнал: амплитуда, период, частота

- **Амплитуда** A – на сколько сигнал отходит от среднего уровня в самой высокой точке. Размах от нижней точки до верхней вдвое больше: $2A$.
- **Период** T – время одного полного колебания, в секундах.
- **Частота** f – сколько колебаний проходит за одну секунду. Единица – герц (Гц): 1 Гц = одно колебание в секунду.

Частота и период – два взгляда на одно и то же. Чем короче период, тем больше колебаний влезает в секунду:

$$f = \frac{1}{T}, \quad T = \frac{1}{f}.$$

Период 0,02 с: $f = 1 : 0,02 = 50$ Гц. Как раз столько в розетке.

Сколько колебаний пройдёт за время t , и наоборот:

$$N = f \cdot t, \quad f = \frac{N}{t}.$$

Струна звучит на 440 Гц. За 0,5 с она качнётся $440 \cdot 0,5 = 220$ раз.

Период по графику. Найди два соседних гребня и вычти их моменты времени. Точнее выйдет, если взять первый гребень и, например, одиннадцатый: между ними 10 периодов, так что разницу времён дели на 10.

Длина волны λ (греческая буква «лямбда») – расстояние, которое волна пробегает за один период:

$$\lambda = v \cdot T = \frac{v}{f},$$

где v – скорость волны. Звук 340 Гц в воздухе: $\lambda = 340 : 340 = 1$ м.

Таблица С.3

Параметр	Что меняется для звука	Что меняется для света
Амплитуда	громкость	яркость

Параметр	Что меняется для звука	Что меняется для света
Частота	высота тона	цвет

Аналоговый сигнал меняется плавно и может принять любое значение: стрелка на старых весах. **Цифровой** – набор отдельных чисел, чаще всего из двух уровней: 0 и 1.

Порог и детектирование

Порог срабатывания – граница, которую сигнал должен превысить, чтобы устройство его заметило. Выше порога – «есть» (1), ниже – «нет» (0). Что делать, если сигнал ровно на пороге, говорит условие.

- **Детектирование по уровню** (амплитудное): смотрим, выше порога сигнал или ниже. Так работает датчик освещённости в уличном фонаре.
- **Детектирование по периоду** (частотное): считаем, как часто сигнал меняется, то есть находим его частоту. Так узнают скорость вращения колеса или высоту звука.

Сколько раз сработал датчик. Считай не точки выше порога, а переходы снизу вверх. Ряд 2, 5, 7, 3, 1, 6, 8, 2 и порог 4: точек выше порога четыре, а срабатываний два – на переходах 2 → 5 и 1 → 6.

Сглаживание: скользящее среднее

Фильтр низких частот (ФНЧ) пропускает медленные плавные изменения и гасит быстрые мелкие скачки – шум. Самый простой ФНЧ на компьютере – **скользящее среднее**:

1. Берём **окно** из k соседних значений (например, три) и находим их среднее.
2. Записываем результат напротив середины окна.
3. Сдвигаем окно на одно значение и повторяем.

Ряд 4, 8, 3, 7, 5 с окном 3: $(4 + 8 + 3) : 3 = 5$, $(8 + 3 + 7) : 3 = 6$, $(3 + 7 + 5) : 3 = 5$. Получилось 5, 6, 5.

- Из n значений окном k выходит $n - k + 1$ средних. Из 5 значений окном 3 – три.
- Чем шире окно, тем глаже линия, но тем больше пропадает мелких деталей.
- **Фильтр высоких частот** (ФВЧ) делает обратное: из сигнала вычитают его сглаженную версию, и остаются только быстрые колебания.

Несущая и огибающая

- **Несущая** – быстрое колебание, которое переносит энергию сигнала на расстояние.
- **Огибающая** – плавная линия через вершины быстрых колебаний. Она медленная, и в ней сидит сама информация.
- Несущая всегда намного быстрее огибающей: за время одного «кусочка» сообщения успевает пройти много её периодов.
- Как приёмник достаёт огибающую: сначала **выпрямляет** сигнал (все отрицательные значения делает положительными), потом **сглаживает** его фильтром низких частот.

Спектр

Сложный сигнал можно сложить из нескольких простых синусоид разной частоты – как белый свет складывается из цветов радуги. **Спектр** – список этих синусоид: какие частоты в сигнале есть и какой у каждой амплитуды. На картинке спектра по горизонтали частота, по вертикали амплитуда, и каждая синусоида – это столбик.

Две синусоиды складываются по точкам: в каждый момент времени значение суммы равно сумме значений. Самая большая высота суммы не больше, чем $A_1 + A_2$.

Для любопытных. Сумма синусоид частотой 100 Гц и 150 Гц повторяется с частотой 50 Гц – это наибольший общий делитель чисел 100 и 150. Значит, период суммы 0,02 с.

Системы счисления

Степени двойки, которые нужны чаще всего:

$$2^0 = 1, 2^1 = 2, 2^2 = 4, 2^3 = 8, 2^4 = 16, 2^5 = 32, 2^6 = 64, 2^7 = 128, 2^8 = 256, 2^9 = 512, 2^{10} = 1024.$$

Из двоичной в десятичную. Подпиши над цифрами справа налево веса 1, 2, 4, 8, 16... и сложи веса там, где стоят единицы: $1011_2 = 8 + 0 + 2 + 1 = 11$.

Из десятичной в двоичную. Дели число на 2 и записывай остатки, пока не дойдёшь до нуля. Остатки читай снизу вверх. Или раскладывай на степени двойки: $45 = 32 + 8 + 4 + 1$, значит $45 = 101101_2$.

Шестнадцатеричная система – шестнадцать цифр: 0–9 и буквы A (10), B (11), C (12), D (13), E (14), F (15). Каждая шестнадцатеричная цифра – это четыре двоичные.

Таблица С.4

Десятичная	Двоичная	Шестнадцатеричная
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B

Десятичная	Двоичная	Шестнадцатеричная
12	1100	C
13	1101	D
14	1110	E
15	1111	F

- **Из шестнадцатеричной в двоичную:** замени каждую цифру её четвёркой.
 $B5_{16} = 1011\ 0101_2$.
- **Из двоичной в шестнадцатеричную:** разбей на четвёрки **справа налево**, слева допиши нули: $101101_2 = 0010\ 1101_2 = 2D_{16}$.
- **Из шестнадцатеричной в десятичную:** старшую цифру умножь на 16 и прибавь младшую: $B5_{16} = 11 \cdot 16 + 5 = 181$.
- Один байт – 8 двоичных цифр или 2 шестнадцатеричные: от 00 до FF, то есть от 0 до 255.

Коды и объём информации

Код – договорённость, какой сигнал что означает. **Алфавит кода** – набор знаков, из которых строят кодовые слова: у двоичного кода это 0 и 1, у азбуки Морзе – точка и тире.

Таблица С.5

Правило	Формула	Пример
Сколько разных слов длины n из k знаков	k^n	3 бита: $2^3 = 8$ кодов
Сколько бит i нужно на M символов	наименьшее i , где $2^i \geq M$	33 буквы: $2^5 = 32$ мало, нужно 6 бит
Объём сообщения из K символов	$I = K \cdot i$	120 символов по 8 бит: 960 бит
Время передачи при скорости v	$t = I : v$	960 бит при 300 бит/с: 3,2 с
Скорость, если один бит длится τ секунд	$v = 1 : \tau$	50 мс на бит: 20 бит/с

Слов длины от 1 до n – сумма по каждой длине. Кодов Морзе длиной от одного до четырёх знаков: $2 + 4 + 8 + 16 = 30$.

Цвет на экране записывают тремя числами от 0 до 255: сколько красного (R), зелёного (G) и синего (B). Три байта – это 24 бита, а значит $2^{24} = 16\ 777\ 216$ разных цветов. В шестнадцатеричной записи цвет пишут как #RRGGBB: #FF0000 – красный, #00FF00 – зелёный, #0000FF – синий, #FFFF00 – жёлтый, #FFFFFF – белый, #000000 – чёрный.

Картинка: объём = число пикселей $\times$ бит на пиксель. Если на пиксель отводят i бит, в палитре 2^i цветов.

Дискретизация – сигнал измеряют через равные промежутки времени. **Частота дискретизации** – сколько измерений (отсчётов) делают за секунду. За время t набирается $f_d \cdot t$ отсчётов, а объём записи звука равен

$$I = f_d \cdot i \cdot t,$$

где i – сколько бит на один отсчёт. Для стереозаписи умножь ещё на 2. Пример: 8000 Гц, 8 бит, 10 с – $8000 \cdot 8 \cdot 10 = 640\,000$ бит, это 80 000 байт.

Для любопытных. Чтобы не потерять колебание, его надо измерить больше двух раз за период: частота дискретизации должна быть больше удвоенной самой высокой частоты сигнала. Это теорема Котельникова. Человек слышит до 20 000 Гц, поэтому на компакт-диске звук записан с частотой 44 100 Гц.

Азбука Морзе

Длительности считают в «точках»: точка – 1, тире – 3, пауза между знаками внутри буквы – 1, между буквами – 3, между словами – 7. В таблице точка – это ., тире – -.

Таблица С.6

Знак	Код	Знак	Код
А	.-	Х	
Б	-...	Ц	-.-.
В	.---	Ч	---.
Г	--.	Ш	----
Д	-..	Щ	--.-
Е, Ё	.	Ъ	--.--
Ж	...-	Ы	-.-
З	--..	Ь	-..-
И	..	Э	..-..
Й	.---	Ю	..--
К	-.-	Я	.-.-
Л	.-..	1	.----
М	--	2	..---
Н	-.	3	...--
О	---	4	-
П	.---	5	
Р	.-.	6	-....
С	...	7	--...

Знак	Код	Знак	Код
Т	-	8	--- ..
У	.. -	9	---- .
Ф	.. - .	0	-----

Пример: буква «С» – ... , это три точки и две паузы внутри буквы: $1 + 1 + 1 + 1 + 1 = 5$ единиц времени.

Модуляция и каналы связи

Канал связи – весь путь сообщения: источник → передатчик (кодирует и модулирует) → среда (воздух, свет, провод; здесь добавляются помехи) → приёмник (детектирует и декодирует) → получатель.

Модуляция – сообщение «сажают» на несущую, меняя один из её параметров:

Таблица С.7

Вид	Что меняется у несущей	Как передать 0 и 1
Амплитудная (АМ)	амплитуда	1 – громко или есть тон, 0 – тихо или тишина
Частотная (ЧМ)	частота	1 – один тон, 0 – другой

Третий параметр, который можно менять, – фаза: сдвиг колебания во времени.

- **Скорость передачи** – сколько бит проходит за секунду, бит/с.
- **Двухканальная передача** – сообщение идёт сразу по двум каналам, например по звуку и по инфракрасному лучу.
- **Бит чётности** дописывают в конец, чтобы число единиц стало чётным. 1011001 – в нём 4 единицы, бит чётности 0, получится 10110010. Если по дороге испортился один бит, единиц станет нечётное число, и приёмник заметит ошибку.
- **Повтор**: каждый бит отправляют три раза, а приёмник выбирает то, чего больше. 110 читается как 1. Одну ошибку в тройке такой код исправляет.
- **Шифр сдвига** (шифр Цезаря): каждую букву заменяют буквой, стоящей в алфавите на k мест дальше; после «Я» снова идёт «А». Расшифровка – сдвиг на k мест назад. Сколько букв в алфавите (32 или 33, с «Ё» или без), смотри в условии.

Логика и комбинаторика

«Истина» – 1, «ложь» – 0.

Таблица С.8

A	B	$A \text{ и } B$	$A \text{ или } B$	НЕ A
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0

A	B	A и B	A или B	НЕ A
1	1	1	1	0

«И» истинно, только когда верны оба условия. «ИЛИ» ложно, только когда неверны оба.

- **Правило умножения:** если выбор идёт по шагам, число вариантов на каждом шаге перемножают. 3 цвета светодиода и 4 уровня яркости – $3 \cdot 4 = 12$ сигналов.
- **Правило сложения:** если выбираем что-то одно из разных групп, варианты складывают.

Проценты, пропорции, уравнения

- **Процент** – сотая часть. 20 % от 350 – это $350 \cdot 0,2 = 70$.
- Какую долю составляет a от b : $a : b \cdot 100$ %. Из 40 отправленных бит испортились 6 – это $6 : 40 \cdot 100 = 15$ %.
- **Пропорция:** если $\frac{a}{b} = \frac{c}{x}$, то $x = \frac{b \cdot c}{a}$. За 4 с прошло 60 бит, а за 10 с – $60 \cdot 10 : 4 = 150$ бит.
- **Среднее арифметическое** – сумма значений, делённая на их количество.
- **Два числа по сумме и разности:** большее = (сумма + разность) : 2, меньшее – сумма минус большее. Сумма 20, разность 4: большее 12, меньшее 8.
- **Уравнение** проверяй подстановкой: найденное число подставь обратно в условие.

Мини-шпаргалки по инструментам

Машина состояний и ПРИМС

ПРИМС – «программирование расширенных иерархических машин состояний». Программу не пишут строчками, а рисуют схемой в Кибериаде IDE: в каких режимах бывает устройство, что его переключает и что оно при этом делает. Потом схему компилируют (превращают в программу для микроконтроллера) и загружают в устройство или в симулятор.

Таблица С.9

Элемент	Что это	Пример для КиберМишки
Состояние	режим работы устройства	«жду», «слушаю», «показываю»
Событие	то, что случилось и заставило реагировать	нажата кнопка, прошла секунда
Условие	фильтр: переход сработает, только если оно верно	громкость выше порога
Действие	что устройство делает	показать картинку на матрице
Переход	смена состояния, на схеме – стрелка	«жду» → «слушаю»

Подпись на стрелке удобно читать так: **событие [условие] / действие**. Например: «пришёл звук [громкость > 50] / показать сердечко».

- В каждый момент машина находится **в одном** состоянии, не в двух сразу.

- Действие на **входе** в состояние выполняется один раз, когда в него попали; на **выходе** – один раз, когда уходят.
- Если условие ложно, перехода нет: устройство остаётся, где было.
- **Иерархия** – это состояние внутри состояния. Стрелка, которая выходит из внешнего состояния, срабатывает, в каком бы из вложенных ни находилась машина.
- Имя состояния отвечает на вопрос «что сейчас происходит?». «Состояние 1» – плохое имя, «жду сигнала» – хорошее.

Как проследить работу машины: заведи таблицу «шаг – текущее состояние – событие – условие верно? – новое состояние – действие» и заполняй строку за строкой. Не прыгай через шаги.

Кибериада IDE: как поставить и где справка

Так было в сентябре 2026 года. Программа обновляется, окна могут поменяться – смысл останется.

- **Где скачать.** Ссылку дают в курсе сферы и в официальной инструкции по установке. По ней лежит актуальная версия для Windows, Linux и macOS.
- **Windows.** Скачай файл с расширением `.exe` и запусти его из папки «Загрузки». Если Windows спросит, можно ли запускать, нажми «Подробнее», а потом «Выполнить в любом случае». Программа может попросить права администратора, чтобы поставить драйверы, – тут нужен взрослый. После установки Кибериада запустится сама, а на рабочем столе появится ярлык.
- **Linux.** Нужны пакеты `libusb`, `udevadm` и `avrdude`. Файл `.deb` – для Ubuntu, Linux Mint и похожих, `.AppImage` и `.snap` – для остальных. Запуск из терминала – командой `cyberiaida-ide`.
- **macOS.** Файл `.dmg`, дальше по подсказкам на экране.
- **Старый КиберМишка версии a7** работает со своей, совместимой версией программы – она лежит в отдельной папке по той же ссылке.
- **Справка.** Нажми **F1**: справа откроется панель справки. Начни с раздела «Быстрый старт – КиберМишка – Начало», а по разделам переходи кнопками внизу панели.
- **Настоящий КиберМишка не нужен:** в Кибериаду встроены симуляторы, на них идут задачи курса и отбора.
- **Если не работает** – вопросы разработчикам пишут в Telegram (t.me/PolyusNT_InstitutLab). Исходный код программы открыт: github.com/polyus-nt/cyberiaida-ide.

Электронные таблицы

Формулы начинаются со знака `=`. В русском Excel и LibreOffice Calc аргументы разделяют точкой с запятой, а дробную часть – запятой.

Таблица С.10

Формула	Что считает
<code>=СРЗНАЧ(В2:В4)</code>	среднее трёх клеток – одно окно скользящего среднего
<code>=МАКС(В2:В50) , =МИН(В2:В50)</code>	наибольшее и наименьшее значение

Формула	Что считает
=СУММ(B2:B50)	сумму
=СЧЁТЕСЛИ(B2:B50;">4")	сколько значений больше 4
=ЕСЛИ(B2>4;1;0)	1, если выше порога 4, иначе 0
=ABS(B2)	модуль: минус убирается, как при выпрямлении
=ОКРУГЛ(C2;1)	округлить до десятых
=ДЕС.В.ДВ(45)	45 в двоичной: 101101
=ДВ.В.ДЕС("101101")	обратно в десятичную: 45
=ДЕС.В.ШЕСТН(181)	181 в шестнадцатеричной: B5
=ШЕСТН.В.ДЕС("B5")	обратно: 181
=2*SIN(2*ПИ()*5*A2)	синусоида с амплитудой 2 и частотой 5 Гц, время – в A2

В английской версии те же формулы называются AVERAGE, MAX, MIN, SUM, COUNTIF, IF, ABS, ROUND, DEC2BIN, BIN2DEC, DEC2HEX, HEX2DEC, SIN, PI, а аргументы разделяют запятой.

- **Скользящее среднее.** Впиши формулу окна в первую клетку и протяни её вниз за угол: адреса сдвинутся сами.
- **График сигнала.** Выдели столбец времени и столбец значений, в меню «Вставка» выбери диаграмму – «график» или «точечную» с линиями.
- **Числа стали текстом?** Если файл с данными пришёл с точками вместо запятых, русский Excel может принять числа за текст или даже за даты. Замени точки на запятые (Ctrl+H) или поменяй разделитель в настройках.

Python (по желанию)

Всё ниже работает в любом онлайн-интерпретаторе Python 3. После # идёт комментарий: Python его не читает.

```

# Системы счисления
print(bin(45))           # 0b101101
print(int("101101", 2))  # 45
print(hex(181))          # 0xb5
print(int("B5", 16))     # 181
print(format(5, "08b"))  # 00000101 - всегда 8 знаков

# Буква → номер в таблице символов → двоичный код
print(ord("A"), format(ord("A"), "08b")) # 65 01000001
print(chr(66))           # B

# Скользящее среднее с окном 3
signal = [4, 8, 3, 7, 5]
k = 3
smooth = []
for i in range(len(signal) - k + 1):
    window = signal[i:i + k]
    smooth.append(sum(window) / k)
print(smooth)            # [5.0, 6.0, 5.0]

# Сколько раз сигнал пересёк порог снизу вверх
data = [2, 5, 7, 3, 1, 6, 8, 2]
porog = 4
count = 0
for i in range(1, len(data)):
    if data[i - 1] < porog <= data[i]:
        count += 1
print(count)             # 2

```

Совет. `bin` и `hex` возвращают строку с припиской `0b` или `0x` впереди. В ответ на отборе её не переписывай. Номера в списке начинаются с 0, а `signal[i:i + k]` берёт элементы с номера `i` до номера `i + k`, не включая последний.

Калькулятор «Программист». В Windows открой встроенный калькулятор, в меню выбери режим «Программист». Строки HEX, DEC, OCT и BIN показывают одно и то же число в шестнадцатеричной, десятичной, восьмеричной и двоичной системах.

Словарик

Таблица С.11

Слово	Что значит
Акселерометр	датчик, который чувствует наклон, толчки и тряску
Алфавит кода	набор знаков, из которых строят кодовые слова: 0 и 1, точка и тире
Амплитуда	на сколько сигнал отходит от среднего уровня в самой высокой точке

Слово	Что значит
Аналоговый сигнал	сигнал, который меняется плавно и может принимать любое значение
Бит	самая маленькая порция информации: один ответ «да» или «нет», 1 или 0
Бит чётности	лишний бит в конце, чтобы число единиц стало чётным; помогает заметить ошибку
Байт	8 бит; в него помещается число от 0 до 255
Датчик	прибор, который превращает свет, звук, температуру в электрический сигнал
Действие	что устройство делает в машине состояний: включить, показать, отправить
Декомпозиция сигнала	разложение сложного сигнала на простые части
Детектирование	выделение полезного из сигнала приёмником: «есть сигнал или нет», «какая частота»
Диаграмма состояний	схема машины состояний: прямоугольники-состояния и стрелки-переходы
Дискретизация	измерение сигнала через равные промежутки времени
ИК-канал	передача сигнала инфракрасным светом, как у пульта от телевизора
Иерархия	в ПРИМС – состояния, вложенные в другое состояние
Канал связи	путь сообщения от передатчика через среду к приёмнику
Кибериада IDE	программа, в которой схемой рисуют ПРИМС и загружают в устройство или симулятор
КиберМишка	программируемое устройство сферы: микрофоны, ИК-датчик, акселерометр, матрица, кнопки
Киберфизическая система	устройства с датчиками и программой, которые сами управляют чем-то в реальном мире
Код	договорённость, какой сигнал что означает
Кодирование и декодирование	перевод сообщения в код и обратно
Компиляция	превращение схемы или текста программы в команды для микроконтроллера
Микроконтроллер	маленький компьютер внутри устройства, который выполняет прошивку

Слово	Что значит
Модуляция	способ посадить сообщение на несущую, меняя её амплитуду, частоту или фазу
Несущая	быстрое колебание, которое переносит сигнал на расстояние
Огибающая	плавная линия через вершины быстрых колебаний; в ней информация
Отсчёт	одно измерение сигнала при дискретизации
Переход	смена одного состояния другим; на схеме – стрелка
Период	время одного полного колебания
Помеха (шум)	всё лишнее, что примешалось к сигналу по дороге
Порог срабатывания	граница, выше которой устройство считает, что сигнал есть
ПРИМС	программирование расширенных иерархических машин состояний
Протокол	правила обмена сообщениями: как начать, в каком порядке слать, как закончить
Прошивка	программа, которую загружают в микроконтроллер
Сигнал	изменение физической величины, которое несёт информацию
Синусоида	самое простое плавное колебание, как качели; её ещё называют гармоническим сигналом
Скользящее среднее	сглаживание: среднее по окну из нескольких соседних значений, окно шагает по ряду
Событие	то, что случилось и может переключить состояние: нажата кнопка, пришёл звук
Состояние	режим, в котором находится устройство
Спектр	список простых колебаний, из которых сложен сигнал, с их частотами и амплитудами
Условие	проверка на переходе: стрелка сработает, только если условие верно
Фильтр низких частот (ФНЧ)	пропускает медленные изменения, гасит быстрые скачки
Фильтр высоких частот (ФВЧ)	пропускает быстрые изменения, убирает медленный фон

Слово	Что значит
Цифровой сигнал	сигнал из отдельных значений, чаще всего из нулей и единиц
Частота	сколько колебаний проходит за секунду; измеряется в герцах
Шифрование	кодирование, чтобы сообщение не прочитал посторонний
ТЮК	Турнир юных киберфизиков – командное инженерное соревнование сферы
ТЮК-Акустика и ИК-ТЮК	комплекты для передачи сигнала звуком и инфракрасным светом

Шпаргалка: всё «Запомни» из глав

Глава 1. Что такое киберфизическая система

Запомни. В КФС всегда есть три роли: датчик, контроллер с прошивкой и исполнитель, и работают они с настоящим миром. Без датчиков система не узнает, что в мире что-то поменялось. Без исполнителей – узнает, но ничего не сможет с этим сделать.

Запомни. Звенья идут так: датчик → передача → обработка → управление → исполнитель, а через мир – снова датчик. Всё, что едет к контроллеру, – информационные сигналы. Всё, что уходит от него к исполнителям, – управляющие.

Запомни. Порог – число, с которым контроллер сравнивает показание датчика. Два порога дают коридор: пока показание внутри него, система ничего не переключает и держит прежнее состояние.

Запомни. Время реакции – сумма всех задержек от события до действия. Считают его по худшему случаю: ожидание опроса – целый период. Путь за это время – скорость в м/с, умноженная на время в секундах.

Запомни. Канал связи: источник → кодирующее устройство → среда → декодирующее устройство → получатель. Управление бывает по плану (вслепую) и с обратной связью (по датчикам). КФС нужно и то и другое: связь доставляет, управление решает.

Глава 2. Сигнал и его параметры

Запомни. Сигнал – изменение величины, в которое вложили смысл. Путь сигнала: источник → среда → приёмник. Контроллер понимает только напряжение, поэтому датчик переводит в него свет, звук и тепло.

Запомни. Аналоговый – плавный, любые значения (пандус). Цифровой – отдельные числа, между ними ничего нет (лестница). Чтобы получить из аналогового цифровой, сигнал измеряют через равные промежутки и округляют до ступеньки.

Запомни. Амплитуда – от среднего уровня до гребня, половина размаха. Период – время одного колебания, частота – сколько колебаний за секунду: $f = 1/T$, и период перед этим переводят в секунды. Громкость – амплитуда, высота тона – частота. Сдвиг на половину периода – противофаза: сигналы гасят друг друга.

Запомни. Отсчёт – одно измерение, дискретизация – измерения через равные промежутки. Частота дискретизации – сколько отсчётов за секунду: $N = f_d \cdot t$. На период сигнала приходится f_d/f отсчётов, и их должно быть больше двух.

Запомни. По уровню – сравниваем с порогом, по периоду – меряем время между повторами. Срабатывание – переход снизу вверх, а не каждая точка выше порога. Время отсчёта = номер · шаг, нумерация с нуля. n щелчков – это $n - 1$ промежутков.

Глава 3. Разложение сигнала: спектр и фильтры

Запомни. Волны складывают по точкам: в каждый момент значение суммы равно сумме значений. Потолок сложной волны – сумма амплитуд, пол – та же сумма с минусом.

Запомни. Осциллограмма – сигнал во времени, спектр – сигнал по частотам. На спектре один столбик – одна простая волна: место столбика – частота, высота – амплитуда. Обертоны музыкального звука стоят на частотах, кратных основной: в 2, 3, 4 раза выше.

Запомни. Медленное – низкие частоты, быстрое – высокие. ФНЧ пропускает то, что ниже частоты среза, ФВЧ – то, что выше. Фильтр смотрит только на частоту волны; её амплитуда ему не важна.

Запомни. Скользящее среднее: окно из k соседних чисел, их среднее – под середину окна, шаг на одно число. Из n чисел выходит $n - k + 1$ средних. Шире окно – глаже линия, но меньше деталей.

Запомни. Несущая быстрая и переносит энергию, огибающая медленная и несёт сообщение. Чтобы достать огибающую, сигнал выпрямляют, потом сглаживают. Колебаний несущей в одном кусочке сообщения: частота несущей $\times$ длительность кусочка.

Запомни. Аномалию находят сравнением: число – с соседями, спектр – со спектром исправной машины, кусок сообщения – с другими копиями. Выброс чистят до сглаживания, иначе он расплзётся.

Глава 4. Кодирование и системы счисления

Запомни. Бит – одна двоичная цифра, байт – 8 бит. В n бит помещается 2^n разных кодов, в байт – 256, это числа от 0 до 255. Сколько бит нужно на M сообщений: найди первую степень двойки, которая не меньше M . Вниз не округляют никогда.

Запомни. Веса двоичных мест справа налево: 1, 2, 4, 8, 16, 32, 64, 128. Из двоичной – сложи веса под единицами. В двоичную – набирай число гирями, начиная с самой тяжёлой, или дели на 2 и читай остатки снизу вверх. Байт – это всегда 8 цифр: недостающие нули пиши слева.

Запомни. $A = 10$, $B = 11$, $C = 12$, $D = 13$, $E = 14$, $F = 15$. Одна шестнадцатеричная цифра – четыре бита, байт – две цифры. Двоичное число режь на четвёрки справа. В десятичную: старшая цифра $\cdot 16$ + младшая; обратно – дели на 16 с остатком.

Запомни. $t = I : v$: биты делим на бит/с, получаем секунды. Один бит длится $1 : v$ секунды. Байты сначала умножь на 8, килобиты в секунду – на 1000. Служебные биты тоже едут по каналу.

Запомни. Бит чётности делает число единиц чётным. На приёме нечётно – ошибка есть, но где она, неизвестно; двух ошибок сразу он не замечает. Контрольная сумма – сумма всех чисел, её правило (например, остаток от деления на 256) всегда даётся в условии.

Запомни. Морзе: точка 1, тире 3, паузы 1 – 3 – 7 (внутри буквы, между буквами, между словами). Чтобы узнать длительность слова, сложи все тоны и все паузы между ними.

Запомни. Равномерный код режут на куски одной длины. Неравномерный читается без разделителей, если ни одно кодовое слово не служит началом другого. Читают его по одному биту: совпало со словом из словаря – отрезай.

Глава 5. Модуляция и каналы связи

Запомни. Модуляция – сообщение «сажают» на несущую, меняя её амплитуду (АМ) или частоту (ЧМ). Демодуляция – приёмник достаёт сообщение обратно. Сообщение делят на кусочки одинаковой длительности τ , по одному биту в каждом.

Запомни. АМ: громко – 1, тихо (или тишина) – 0. Запись режут на кусочки по τ и в каждом смотрят размах. Длинная пачка – несколько одинаковых бит подряд: длительность пачки делим на τ . Колебаний несущей в одном бите: $n = f \cdot \tau$.

Запомни. ЧМ: высокий тон – одно значение бита, низкий – другое, какое именно – договариваются заранее. Частота по графику: $f = n : \tau$ или $f = 1 : T$. ЧМ меньше боится расстояния и шума, чем АМ. Частоты выбирают внутри рабочего диапазона, вдали от шума и друг от друга.

Запомни. Канал: передатчик → среда → приёмник. Звук: 340 м/с, огибает преграды, боится шума и эха. ИК: почти мгновенно, нужна прямая видимость, мешают солнце и лампы. Задержка в пути $t = s : v$.

Запомни. $t = I : v$, где I – все биты вместе со стартовой меткой, паузами и проверочными битами. Доля ошибок = ошибочные биты : отправленные · 100 %. Повтор трижды с голосованием исправляет одну ошибку в тройке, но удлиняет сообщение втрое.

Запомни. Два канала сразу дают на выбор надёжность (одно и то же дважды) или скорость (сообщение поделили). Два передатчика на разных частотах разделяет спектр, и у каждого свой порог. Время поделённой передачи – по тому каналу, который закончит позже.

Глава 6. ПРИМС: машины состояний в Кибериаде

Запомни. Состояние – это режим, в котором устройство сейчас. Машина всегда в одном состоянии и остаётся в нём, пока её не переключат. Чёрная точка показывает, откуда машина начинает.

Запомни. Подпись стрелки: событие [условие] / действие. Событие ищут только среди стрелок из текущего состояния. Нет стрелки или условие неверно – машина стоит на месте. Действие выполняется при переходе.

Запомни. Трассировка: состояние – событие – есть ли стрелка из **этого** состояния – верно ли условие – новое состояние. Одна строка на событие, без прыжков через шаги. Таймер считает с момента входа в состояние.

Запомни. Вложенное состояние первым отвечает на событие. Нет у него своей стрелки – отвечает рамка. Вошёл в рамку – попал в её начальное состояние. Одна стрелка от рамки заменяет по стрелке из каждого вложенного состояния.

Запомни. Тупик – нет выхода. Недостижимое – нет пути из начального (проверяй обводкой, а не поиском входящей стрелки). Две стрелки с одним событием из одного состояния – условия не должны пересекаться.

Глава 7. Управление КиберМишкой

Запомни. Исполнитель выполняет только команды из своего списка. У КиберМишки датчики присылают **события** и **значения**, а глаза, матрица, пищалка и ИК-светодиод выполняют **действия**. Садовник: «Вперёд» и «Назад» – на клетку без поворота, повороты – на месте; на север y уменьшается.

Запомни. Порядок придумывания: режимы → что их переключает → что делать при входе. Событие без своей стрелки из текущего состояния пропадает. Таймер Садовника срабатывает один раз и запускается заново, таймер КиберМишки (по справке) тикает, пока его не остановят.

Запомни. Шкала датчика 0–4095 – это 12 бит. Событие по порогу – это переход через порог, а не каждый отсчёт над ним. «Интервал срабатывания» глушит эхо. Остаток от деления байта на 4 – две последние двоичные цифры.

Запомни. Трассировка – таблица, строка за строкой. Направо – по часовой стрелке $C \rightarrow B \rightarrow Y \rightarrow 3$. Меньше команд – значит, меньше поворотов. Клеток в рамке поля $W \times H$: $2 \cdot (W + H) - 4$.

Глава 8. Математика киберфизика

Запомни. Сначала скажи формулу словами, потом подставляй. Числа – в тех единицах, в которых записана формула: герцы, секунды, биты. Черта дроби делит и одновременно работает как скобки.

Запомни. К более мелкой единице – умножаем, к более крупной – делим. Кило – тысяча, мега – миллион, милли – тысячная, микро – миллионная. 1 байт = 8 бит. Связь – по 1000 (кбит/с), память – по 1024 (Кбайт).

Запомни. Долю в проценты – умножить на 100. «На сколько процентов» – от того, что было до. Прямая пропорция: больше времени – больше бит во столько же раз. Обратная: скорость выше в 3 раза – время меньше в 3 раза.

Запомни. С обеими частями уравнения делай одно и то же. Два неизвестных – нужны два уравнения. Одинаковые знаки у неизвестного – вычитай уравнения, разные – складывай. Ответ всегда проверяй подстановкой в оба уравнения.

Запомни. И – нужны оба, ИЛИ – хватит одного (и оба тоже годятся), НЕ – наоборот. У n входов таблица истинности из 2^n строк, пиши их двоичным счётом. Порядок: скобки, НЕ, И, ИЛИ.

Запомни. По шагам – умножай, по отдельным группам – складывай. Повторы разрешены – n^k . Повторы запрещены – каждый следующий множитель на единицу меньше. Запрет на первом месте уменьшает только первый множитель.

Запомни. Формула начинается с «=». При протягивании обычные адреса едут, а адреса с долларами стоят на месте. Пустая клетка – это ноль. Общий для всех строк порог держи в отдельной клетке и ссылайся на неё через \$.

Где что найти

Таблица С.12

Что	Где
Правила олимпиады, сферы, сроки	junior.ntcontest.ru
Личный кабинет участника	my.ntcontest.ru
Курс сферы «Кибертех» (за тест +5 баллов к отбору)	academy.junior.online/event/cybertech_junior , вход через «Талант»
Платформа отбора	orbita.education (заходить кнопкой из личного кабинета)
Программа кружка и оборудование сферы	junior.online
Памятки для Новосибирской области	nto.kvantorium54.ru
Кибериада IDE: исходный код и вопросы	github.com/polyus-nt/cyberiada-ide , t.me/PolyusNT_Insitulab
Разбор отбора 2025 года (сфера тогда – «Технологии и киберфизические системы»)	vkvideo.ru/video-187440231_456239186
Журнал «Юный киберфизик»	journal.cyberphys.ru
Новости НТО Юниоры	vk.ru/ntojunior

Об авторе-составителе



Артём Осиновский

Методист и разработчик детского технопарка «Кванториум» (Новосибирск). Придумывает образовательные программы и сам делает для них цифровые инструменты: платформу смены «Миссия на Луну» для фестиваля «Технопром-2026», сайт с памятками НТО Юниоры для Новосибирской области. Эту книгу собрал для школьников области, чтобы к отбору можно было подготовиться самостоятельно – даже без кружка рядом с

домом.

Нашли ошибку – пишите на osinovsky.aa@kvantorium54.ru (рабочая почта): сфера, версия книги и страница. По другим вопросам и проектам – личная: artem081033@gmail.com.