ЗОЛОТОЕ СЕЧЕНИЕ
ФОНД ПОДДЕРЖКИ ТАЛАНТЛИВЫХ ДЕТЕЙ И МОЛОДЕЖИ

ВСОШ

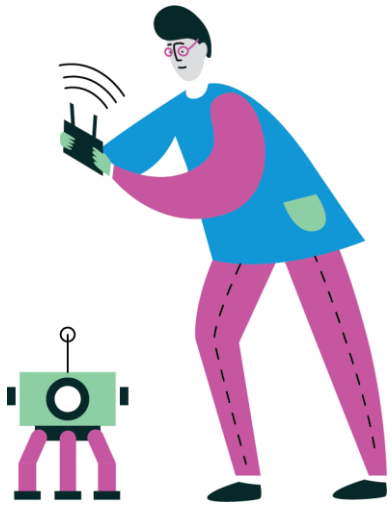
Разбор заданий муниципального этапа всероссийской олимпиады школьников по труду (технологии), робототехника 9 класс

2024/2025 учебного года в Свердловской области

Разработчик –
Гущин Леонид Олегович

Задание 1-5

1. Генная инженерия
2. А
3. Специалист по 3D-печати (Инженер по 3D-печати)
4. В
5. Б

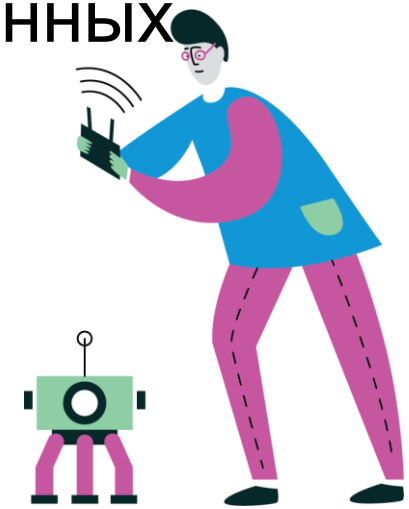


Задание 6

Arduino uno можно запитать через USB порт напряжением 5V либо через специальный разъём от источника 7-12 вольт

Ток потребления двух светодиодов меньше тока, который выдают все перечисленные источники питания.

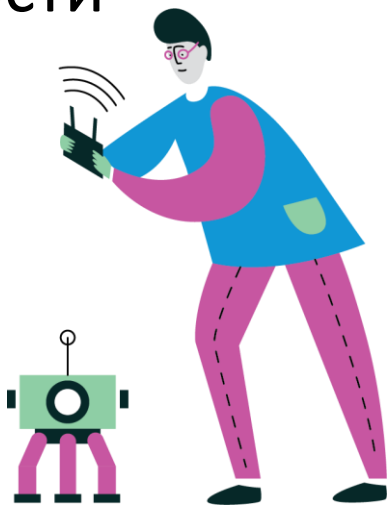
Соответственно можно питать от любого из перечисленных источников.



Задание 7

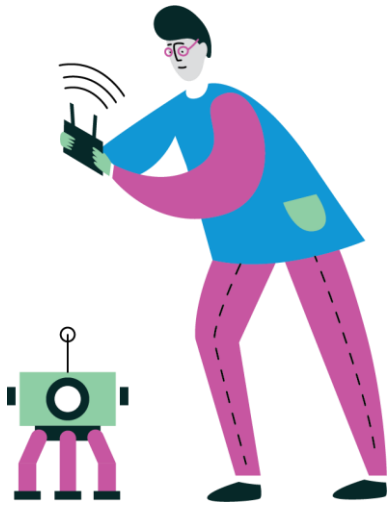
Согласно ГОСТ Р 60.0.0.4-2019

2.8 робототехническое устройство (robotic device): Исполнительный механизм, обладающий характеристиками промышленного робота (2.9) или сервисного робота (2.10), но не имеющий либо необходимого числа программируемых степеней подвижности (4.3). либо некоторой степени автономности.



Задание 8

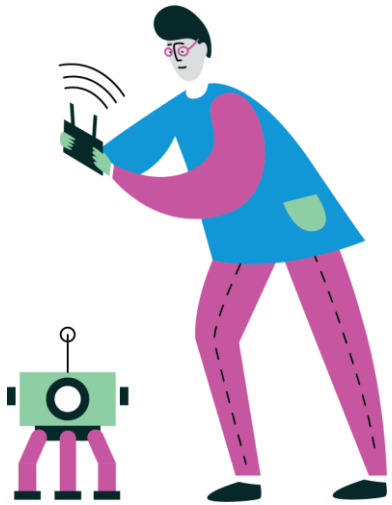
DC — это аббревиатура «Direct Current», что означает постоянный ток. Соответственно DC motor – это электродвигатель постоянного тока.



Задание 9

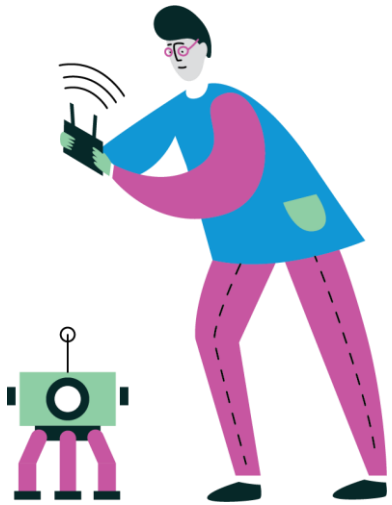
Классическое определение нейрона в нейросетях:

Нейрон это элементарная вычислительная единица, которая получает информацию, производит вычисления и передаёт данные дальше.



Задание 10

Червячная передача может передавать движение только от червяка к зубчатому колесу и таким образом нельзя поменять местами ведущую и ведомую оси.



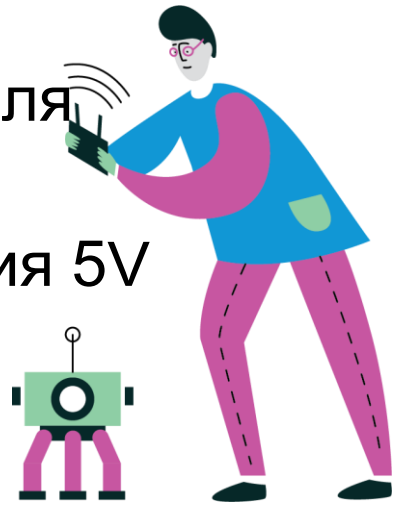
Задание 11

Сервопривод sg90 в движении потребляет до 80мА, соответственно 12 сервоприводов потребляют до 960мА

USB порт компьютера выдаёт ток от 500мА (версия 2) до 900мА (версия 3) соответственно для питания не подходит.

Питать через порт USB Arduino не рекомендуется при сборке силовых схем, так как там стоит специальный диод который при любых неполадках перегревается и может выйти из строя, его ставят для защиты USB порта компьютера от коротких замыканий.

Соответственно лучший вариант – запитать от источника питания 5V напрямую, не забыв объединить GND устройств в схеме.



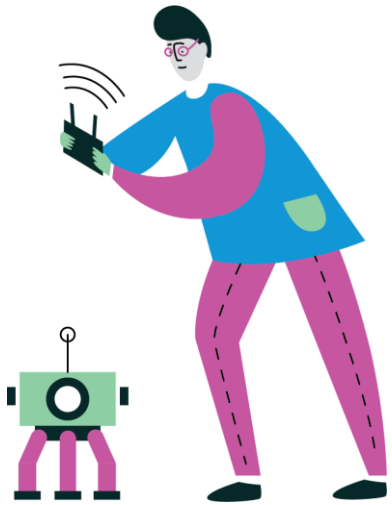
Задание 12

Так как датчик при срабатывании выдаёт 0, общий принцип работы:

Если все 1 схема даёт 0

Если хотя бы один 0, схема даёт 1

Данному принципу соответствует только одна операция, И-НЕ



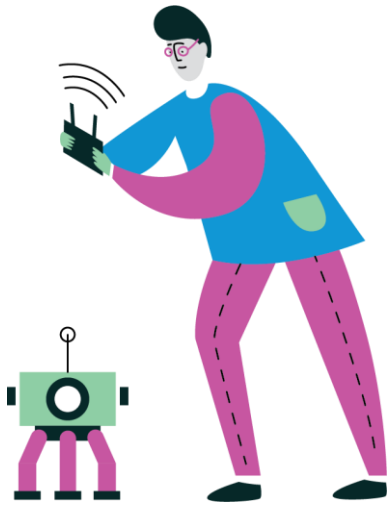
Задание 13

Рассчитаем общий потребляемый ток, переведя в единицу измерения Ампер:

0,2 + 6 * 0,8 = 0,2 + 4,8 = 5А

Мощность рассчитывается как произведение силы тока на напряжение:

5 * 12 = 60 Ватт

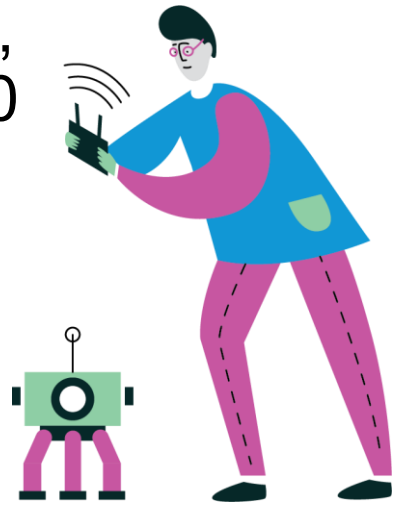


Задание 14

Передаточное отношение в ременной передаче рассчитывается как отношение диаметров ведущего и ведомого шкивов. Соответственно 10/60 = 1/6

Соответственно для одного оборота ведомого шкива надо сделать 6 оборотов ведущего шкива, то есть соотношение для передачи 1 к 6.

Таким образом если ведущий шкив повернулся на 360 градусов, ведомый даст угол поворота в 6 раз меньше, то есть 360 / 6 = 60 градусов



Задание 15

Рассмотрим первые два зубчатых колеса в системе. Передаточное отношение равняется 8 / 40 = 1 к 5.

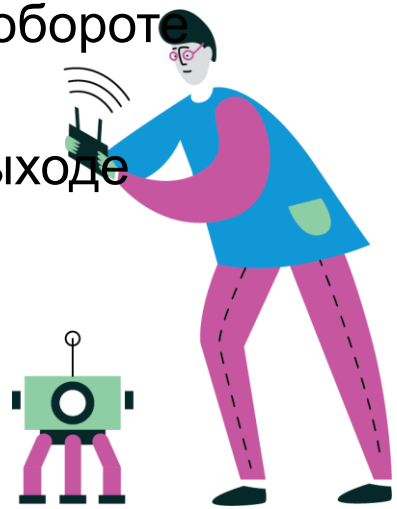
Соответственно если на входе сделали 200 оборотов, на выходе передачи будет

200 / 5 = 40 оборотов

Червяк находится на этой же оси, соответственно тоже сделает 40 оборотов

Зубчатое колесо присоединенное к червяку сдвигается на 1 зуб при полном обороте червяка, соответственно делает один оборот на 8 оборотов червяка.

Таким образом получаем что при 40 оборотах червяка зубчатое колесо на выходе сделает 40 / 8 = 5 оборотов.



Задание 16

Для вычисления количества информации используется формула:

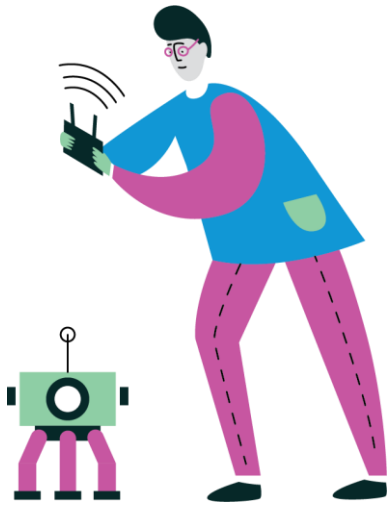
$N = 2^i$

Всего у нас 4096 значений (от 0 до 4095) Тогда получается уравнение:

$4096 = 2^i$

$2^{12} = 2^i$

$i = 12$



Задание 17

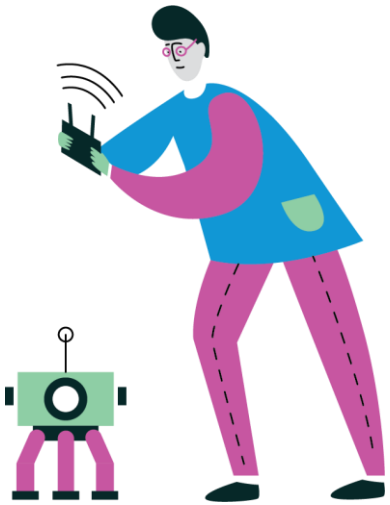
Робот за 1 оборот колеса проезжает расстояние равное длине окружности, рассчитаем это расстояние используя формулу длины окружности:

$S = 2* \pi * r = 2 * 3.14 * 80 = 502,4$мм

Поделим расстояние, которое нужно проехать роботу (переведя его из метров в миллиметры) на это число:

1000 / 502,4 = 1.99

Округлим полученное значение до ближайшего целого, получим 2 оборота.



Задание 18

Если робот поворачивает одним колесом, это колесо проходит траекторию представляющую собой дугу окружности, радиус которой равен расстоянию между колёсами. Для поворота на угол 90 градусов роботу необходимо повернуться на четверть данной окружности. Рассчитаем, какую длину траектории необходимо пройти правому колесу:

$S = (2 * \pi * r) / 4 = (2 * 3.14 * 200) / 4 = 314$ мм

Рассчитаем, какое расстояние проезжает колесо за 1 оборот:

$S = 2 * \pi * r = \pi * D = 3.14 * 50 = 157$мм

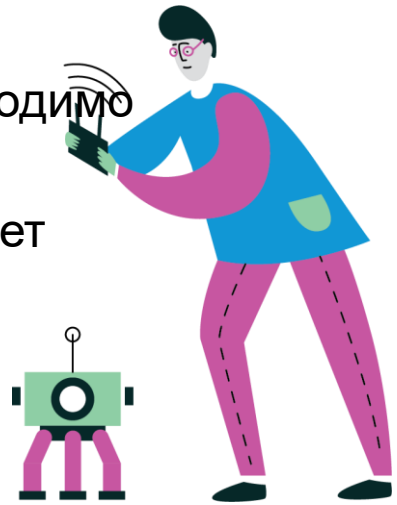
Соответственно для того чтобы колесо проехало необходимые 314 мм, роботу необходимо сделать 314 / 157 = 2 оборота

Энкодер считается каждые 10 градусов, соответственно за 1 оборот энкодер насчитает

360 / 10 = 36 значений

За два оборота которые необходимо проехать энкодер насчитает

36 * 2 = 72 значения



Задание 19

Рассмотрим условие цикла:

for(int i = 10; i > 3; i = i - 2)

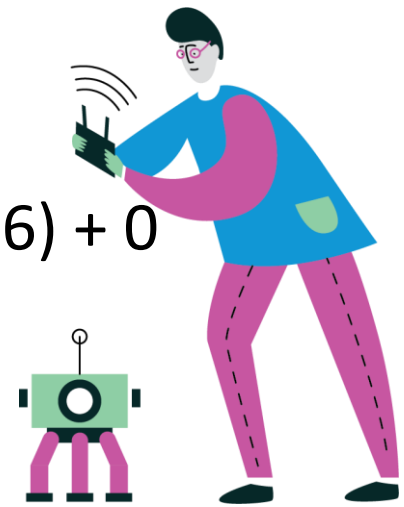
В соответствии с условием i будет принимать следующие значения:

10, 8, 6, 4

Начальное значение sum равно 10, к нему прибавляются значения считанные с портов, номер которых указывает i, соответственно итоговый результат будет:

10 (начальное значение) + 1 (с порта 10) + 1 (с порта 8) + 0 (с порта 6) + 0 (с порта 4)

10 + 1 + 1 + 0 + 0 = 12



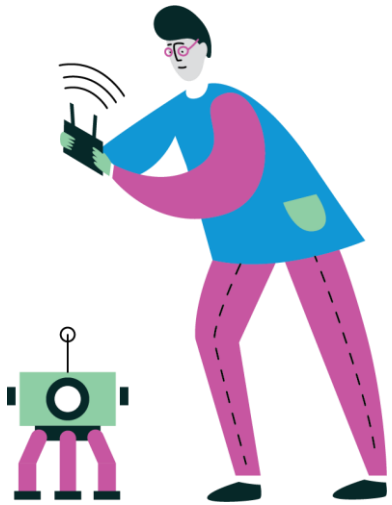
Задание 20

Команда pinMode(11, INPUT_PULLUP); конфигурирует порт 11 на чтение с подтягивающим резистором. Это значит что при ненажатой кнопке мы получаем при чтении 1 от подтягивающего резистора, при нажатой кнопке 0.

Щелчок кнопкой – это последовательное нажатие а затем отпускание кнопки. Соответственно чтобы обработать щелчок мы должны постоянно опрашивать кнопку ожидая 0 (нажатие) затем 1 (отпускание). При этом для постоянного опроса if не подходит, так как делает только одну проверку.

Соответственно правильным алгоритмом ожидания щелчка будет:

```
while(digitalRead(11) == 0);
while(digitalRead(11) == 1);
Serial.println(“click”);
```



Задание 21

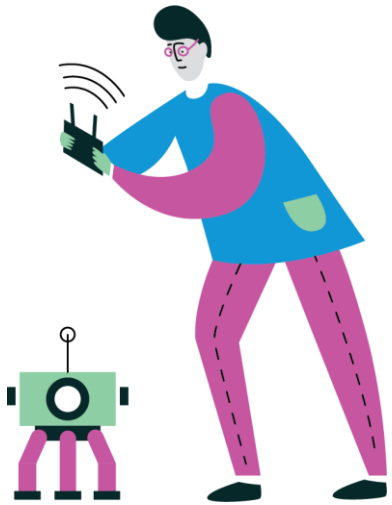
Входной диапазон от 0 до 4095. Найдём шаг выходного диапазона относительно входного:

4095 / 100 = 40,95

То есть на каждые 40,95 значений входного диапазона будет прибавляться единица на выходе.

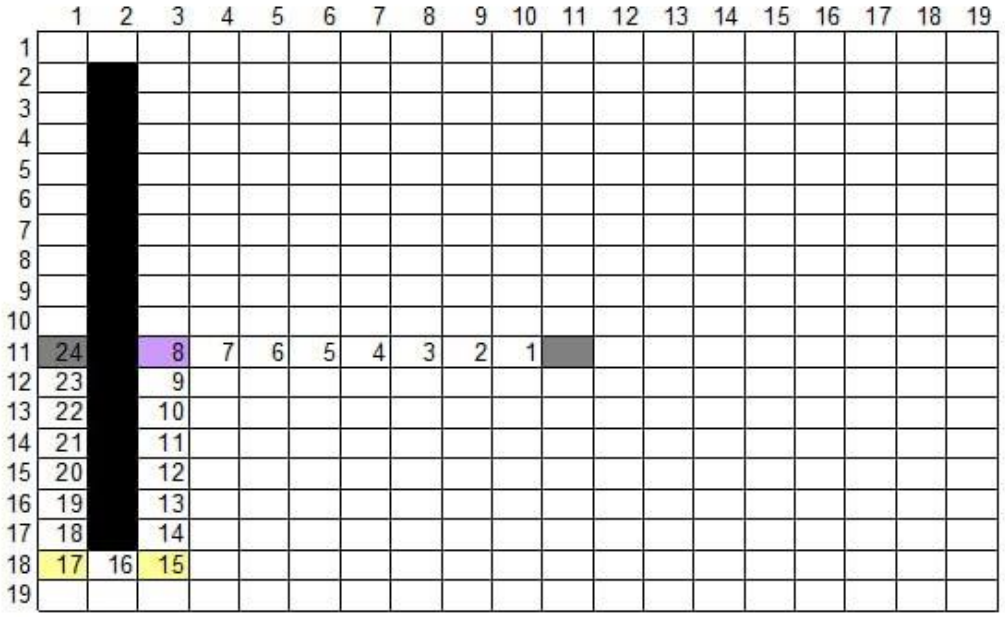
Тогда числу 615 соответствует:

615 / 40, 95 = 15 (округляем до целого в соответствии с условием задачи)



Задание 22

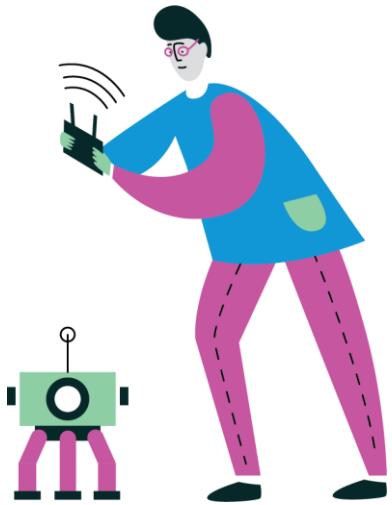
Проиллюстрируем решение задачи:



24*200=4800 проехал прямо
минус 200 поворот налево
плюс 200 плюс 200 поворот направо

поворот налево
поворот направо

итого: 4800-200+200+200 = 5000



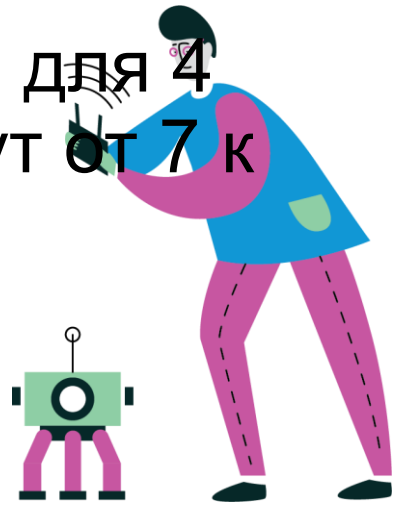
Задание 23

Красный светодиод на 6 порту, жёлтый на 4 порту. Соответственно 6 и 4 бит необходимо установить в 1 остальные не изменять. Побитовое ИЛИ на 1 изменяет либой бит устанавливая в 1 побитовое ИЛИ на 0 не меняет значение бита.

Соответственно надо сделать побитовое ИЛИ указав 1 для 4 и 6 бита, учитывая что биты в последовательности идут от 7 к 0 слева направо:

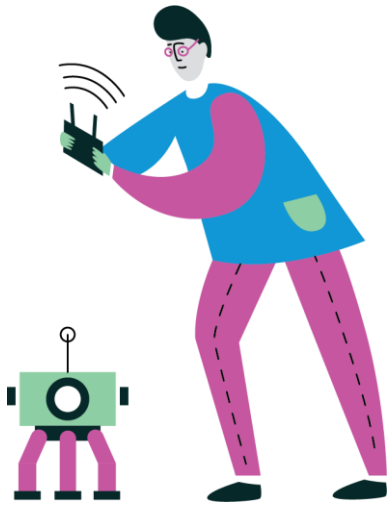
```
PORTD |= B01010000
```

Ответ: 01010000



Задание 24

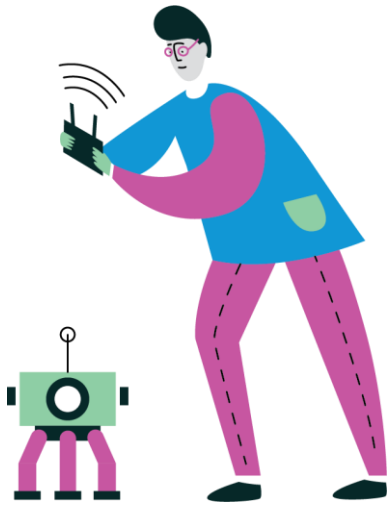
Функция millis () возвращает количество миллисекунд, прошедших с момента запуска программы на Arduino. Она является полезным инструментом для отслеживания времени в вашем проекте. Возвращаемое значение имеет тип unsigned long, что позволяет отслеживать время на протяжении более 49 суток.



Задание 25

Данная задача классическая для использования математической модели для вычисления относительно переменной итератора i различных значений. Для построения модели рассмотрим как должно вычисляться итоговое значение с использованием этой переменной:

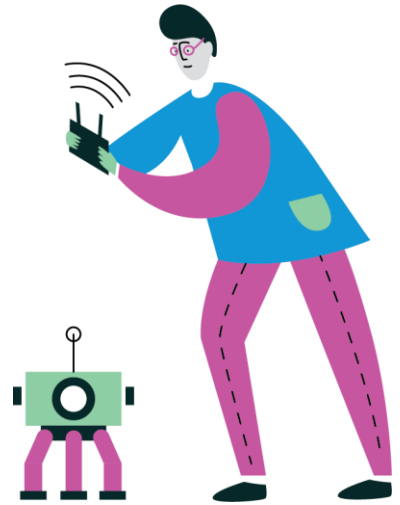
i	Значение 11 порта	Значение 12 порта
0	0	255
1	1	254
2	2	253
3	3	251
…	…	…
254	254	1
255	255	0



Проанализировав эти данные мы увидим, что сумма значений 11 и 12 порта всегда равняется 255. Соответственно на 11 порту значение всегда равняется i. На 12 порту таким образом значение которое мы рассчитываем через итератор i будет равняться 255 - i.

Практическое задание

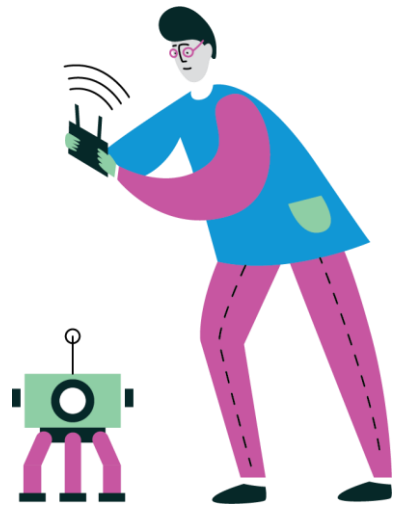
Задание сформулировано так, что каждая подзадача решается не одним а сразу несколькими способами. Мы сформулируем для решения отдельных задач разные способы их решения, для остальных наиболее оптимальные.



Практическое задание

Задача 1. Смена состояний устройства.
Наилучший способ решения этой задачи - специальная переменная - флаг, которая хранит в себе текущий режим.
Например 0 - состояние готовности устройства
1 - состояние счёта времени
2 - состояние паузы
Тогда общая структура программы будет выглядеть так:

```
byte state = 0;
void loop() {
  if(state == 0){
    //обработка состояния готовности
  }
  if(state == 1){
    //обработка состояния счёта времени
  }
  if(state == 2){
    //обработка состояния паузы
  }
}
```

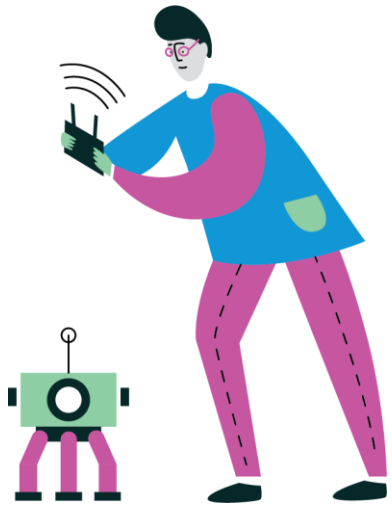


Практическое задание

Задача 2. Проверка нажатий на кнопку.

Данная задача решается разными способами. Самый простой - проверка кнопок прямо внутри обработчиков состояний, для примера рассмотрим обработчик состояния готовности (считаем что кнопки на 2 и 3 портах). Кнопки сконфигурированы подтягивающим резистором, то есть при ненажатом состоянии дают 1, нажатом 0. Это достигается командой pinMode(2, INPUT_PULLUP) в разделе setup, либо сборкой схемы кнопки с подтягивающим резистором:

```
if(state == 0){
    if(digitalRead(2) == 0 && digitalRead(3) == 1) {
      delay(100);
      if(digitalRead(2) == 0 && digitalRead(3) == 1){
        state = 1;
      }
   }
}
```



Здесь важно проверить что нажата только верхняя кнопка, так как если нажаты обе либо обе не нажаты, либо нажата нижняя - во всех этих случаях сохраняется состояние готовности.

В остальных ветках проверка будет аналогичной, но необходимо проверять несколько комбинаций.

Основной минус такой проверки - программа блокируется на 100мс, что может привести к неправильному подсчёту времени если считать время через delay а не через millis (см задачу счёт времени)