

Цель работы: Игра с кнопками на Arduino под названием "Угадай код" является простой и увлекательной игрой, где целью игрока является угадать определенный код, представленный в виде последовательности нажатий на кнопки.

Цель игры заключается в том, чтобы правильно воспроизвести заданный код, используя правильную последовательность нажатий на кнопки. Каждая кнопка будет представлять определенное значение или символ, и игрок должен будет угадать и воспроизвести правильную последовательность нажатий, чтобы разгадать код. Обычно игра предлагает игроку некоторую начальную последовательность, и игроку необходимо повторить эту последовательность точно так же, чтобы пройти на следующий уровень. С каждым уровнем последовательность становится длиннее или усложняется, что делает игру более интересной и вызывает необходимость концентрации и запоминания.

Цель игры "Угадай код" заключается в достижении максимального уровня, проходя все более сложные последовательности и угадывая коды. Эта игра развивает навыки памяти, внимания и реакции, а также предоставляет увлекательный способ провести время с Arduino.

### **Список компонентов :**

- 1.Arduino Nano
- 2.Тактовые кнопки (3-штуки)
- 3.Светодиода (2 штуки, «красный и зелёный»)
- 4.Резисторы (4 штуки, 220 Ом)
- 5.Макетная плата
- 6.Провода «папа-папа и мама-папа»

1. Arduino Nano - это компактная плата разработки, основанная на микроконтроллере ATmega328P. Она представляет собой миниатюрную версию платы Arduino Uno и обладает похожими функциями и возможностями.

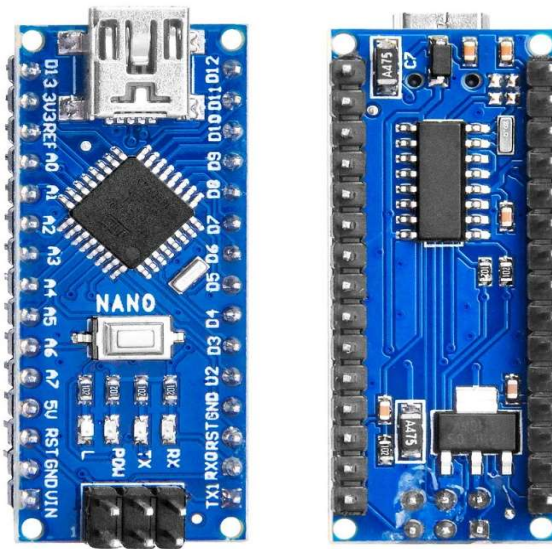


Рисунок 1 Arduino Nano

Arduino Nano имеет следующие характеристики:

1. Микроконтроллер: ATmega328P с тактовой частотой 16 МГц.
2. Входное напряжение: 7-12 В.
3. Цифровые входы/выходы: 14 (из которых 6 могут быть использованы как ШИМ-выходы).
4. Аналоговые входы: 8.
5. Ток на выводе пина: 40 мА.
6. Память: 32 КБ флеш-памяти (включая 2 КБ, используемые загрузчиком), 2 КБ ОЗУ и 1 КБ EEPROM.
7. Интерфейсы: UART, SPI, I2C.
8. USB-интерфейс: для программирования и подключения к компьютеру.

Arduino Nano обладает маленьким размером и удобными пин-хедами, что делает ее идеальной для проектов, где ограниченное пространство является фактором. Она может быть использована для различных целей, включая создание интерактивных устройств, роботов, систем автоматизации, измерительных приборов и других электронных проектов.

## 2. Тактовая кнопка

Тактовая кнопка, также известная как кнопка с механическим контактом или моментальная кнопка, является одним из наиболее распространенных типов кнопок, используемых в электронных устройствах. Вот некоторая информация о тактовых кнопках



Рисунок 2 Тактовая кнопка

1. Механизм работы: Тактовая кнопка состоит из корпуса, контактных элементов и пружины. При нажатии на кнопку контактные элементы замыкаются, создавая электрическое соединение и передавая сигнал о нажатии.
2. Однократное действие: Тактовая кнопка обеспечивает мгновенное (однократное) действие. Она открывает или закрывает электрическое соединение при каждом нажатии и возвращается в исходное положение после отпускания.
3. Надежность контактов: Контакты в тактовой кнопке могут быть выполнены из различных материалов, таких как медь, серебро или золото, для обеспечения надежного электрического контакта. Это помогает предотвратить дребезг контактов (эффект многократного срабатывания) при нажатии на кнопку.

4. Форм фактор и дизайн: Тактовые кнопки доступны в различных формах и размерах. Они могут быть с монтажом на плату (SMD) или с пинами для проводного подключения. Корпус кнопки может быть прозрачным или непрозрачным, с различными цветами и маркировкой.

3. Светодиод (LED, от англ. Light-Emitting Diode) - это полупроводниковое устройство, которое излучает свет, когда через него пропускается электрический ток. Светодиоды являются одними из наиболее распространенных и популярных электронных компонентов, используемых в различных приложениях.



Рисунок 3 Светодиод

Особенности светодиодов:

1. Энергоэффективность: Светодиоды потребляют меньше энергии, чем традиционные источники света, такие как галогенные лампы или люминесцентные лампы. Они обладают высоким КПД и могут производить большой световой поток при небольшом энергопотреблении.
2. Долговечность: Светодиоды обладают длительным сроком службы. Они имеют значительно большую продолжительность работы по сравнению с традиционными источниками света. Обычно светодиоды могут работать от нескольких тысяч до нескольких десятков тысяч часов.
3. Быстрый отклик: Светодиоды могут мгновенно включаться и выключаться. Они не требуют времени прогрева или охлаждения и могут быстро менять свое состояние в ответ на изменения в подаче электрического тока.

4. Малые размеры: Светодиоды имеют компактный размер, что делает их удобными для использования в различных проектах и приложениях. Они могут быть интегрированы в малогабаритные устройства и иметь различные формы и размеры.
5. Широкий цветовой спектр: Светодиоды доступны в различных цветах, включая красный, зеленый, синий, желтый, оранжевый и белый. Также существуют RGB-светодиоды, которые могут создавать широкий спектр цветов путем комбинации основных цветов.

4. Резисторы - это электронные компоненты, предназначенные для ограничения потока электрического тока в электрической цепи. Они имеют определенное сопротивление, которое измеряется в омах ( $\Omega$ ). Резисторы широко используются в электронике для контроля тока, создания делителей напряжения, фильтрации сигналов и других приложений.



Рисунок 4 Резисторы

Особенности резисторов:

1. Сопротивление: Резисторы предоставляют определенное сопротивление электрическому току. Они могут иметь фиксированное (непеременное) сопротивление или быть переменными, с возможностью изменения сопротивления.
2. Номинальное значение: Резисторы имеют указанное номинальное сопротивление, которое обозначается на их корпусе или в технической документации. Номинальное значение представляет собой предполагаемое сопротивление резистора, с учетом допустимых отклонений.
3. Точность: Резисторы могут иметь различную точность, выраженную в процентах. Точность определяет, насколько близко реальное сопротивление резистора к его номинальному значению.

4. Мощность: Резисторы имеют определенную мощность, которая указывает на максимальную мощность, которую они могут рассеивать без перегрева. Мощность резистора влияет на его физический размер.
5. Температурный коэффициент сопротивления: У некоторых резисторов сопротивление может изменяться в зависимости от температуры. Температурный коэффициент сопротивления характеризует это изменение.

5. Макетная плата (также известная как плата для прототипирования или breadboard) - это основа, используемая для создания временных электрических соединений между электронными компонентами. Она представляет собой пластиковую или фенолформальдегидную плату с рядами отверстий, в которые можно вставлять электронные компоненты и провода для создания цепей и схем.

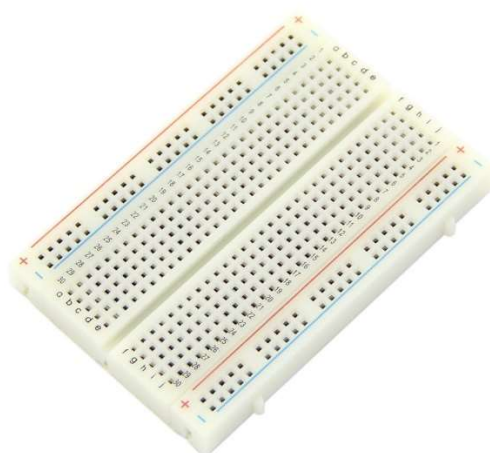


Рисунок 5 Макетная плата

Особенности макетных плат:

1. Отверстия: Макетные платы имеют сетку отверстий, расположенных в рядах и столбцах. Каждое отверстие представляет собой точку контакта, в которую можно вставить ножки компонентов или провода для установки соединений.
2. Расположение контактов: В большинстве макетных плат контакты в каждом ряду горизонтально соединены внутри платы, тогда как контакты в разных рядах

вертикально разделены. Это позволяет создавать электрические соединения между компонентами и проводами, вставленными в разные ряды.

2.Размеры и типы: Макетные платы доступны в разных размерах и конфигурациях. Существуют маленькие платы для небольших проектов, а также более крупные, предназначенные для сложных схем.

#### 6.Провода «папа-папа», «папа-мама»

Провода "папа-папа" и "папа-мама" являются распространенными терминами, используемыми для описания типов разъемов и соединительных кабелей.



Рисунок 6 Провода «папа-папа», «папа-мама»

Провод "папа-папа" (также известный как мужской-мужской) имеет на обоих концах разъемы, которые называются мужскими разъемами. Мужской разъем обычно имеет выступающий пин или штырь, который вставляется в соответствующий разъем с отверстием или гнездом. Примеры применения провода "папа-папа" включают подключение компьютера к монитору с помощью VGA-кабеля или подключение аудиоустройств с помощью аналогового аудиокабеля с двумя мужскими разъемами.

Провод "папа-мама" (также известный как мужской-женский) имеет разъемы разных полов - мужской разъем на одном конце и женский разъем на другом конце. Женский разъем обычно имеет отверстие или гнездо, в которое вставляется пин

или штырь мужского разъема. Примеры применения провода "папа-мама" включают подключение периферийных устройств, таких как клавиатура или мышь, к компьютеру с помощью USB-кабеля или подключение мобильного устройства к зарядному устройству с помощью кабеля с мужским USB-разъемом и женским разъемом для зарядки.

## 7.Схема подключения

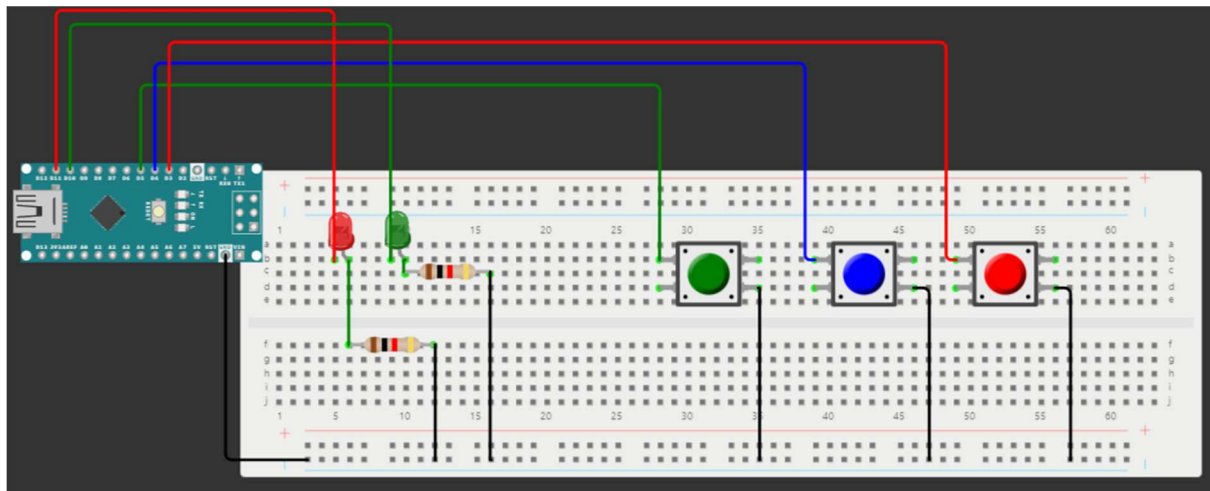


Рисунок 7 Схема подключения

Таблица 1

|                      |                     |
|----------------------|---------------------|
| Arduino Nano (pin12) | Светодиод “Красный” |
| Arduino Nano (pin10) | Светодиод “Зелёный” |
| Arduino Nano (pin 5) | Тактовая кнопка     |
| Arduino Nano (pin 4) | Тактовая кнопка     |
| Arduino Nano (pin 3) | Тактовая кнопка     |
| Arduino Nano (GND)   | breadboard          |



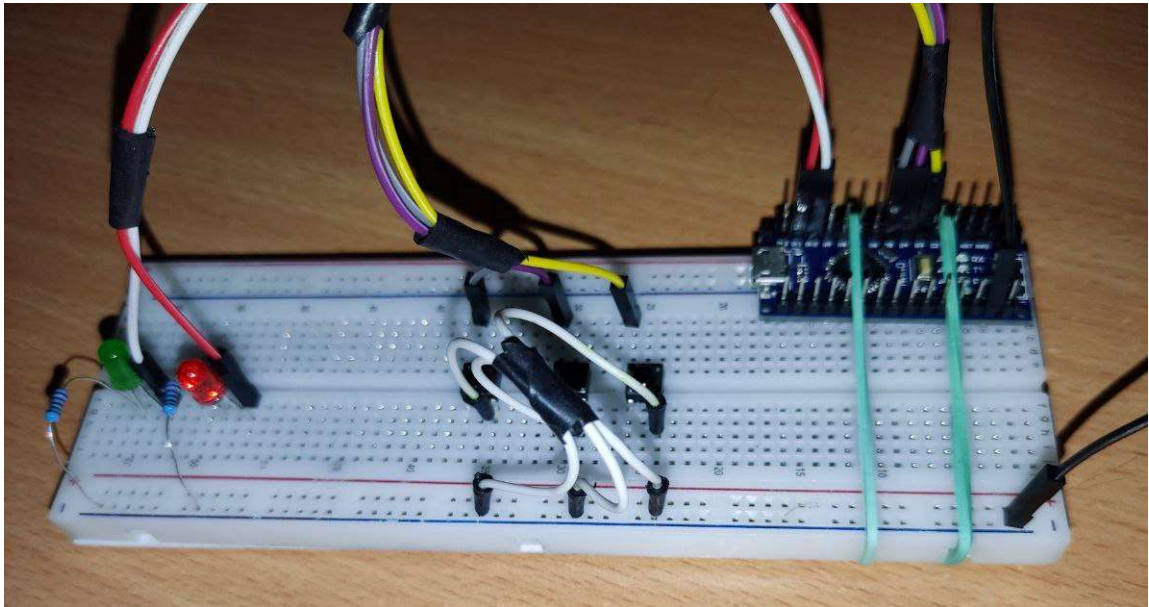


Рисунок 8 Схема подключения

## 8. Код на (C++ на фреймворк Wiring)

1. Массив с вариантами кода, и переменные для считывания нажатия на кнопки

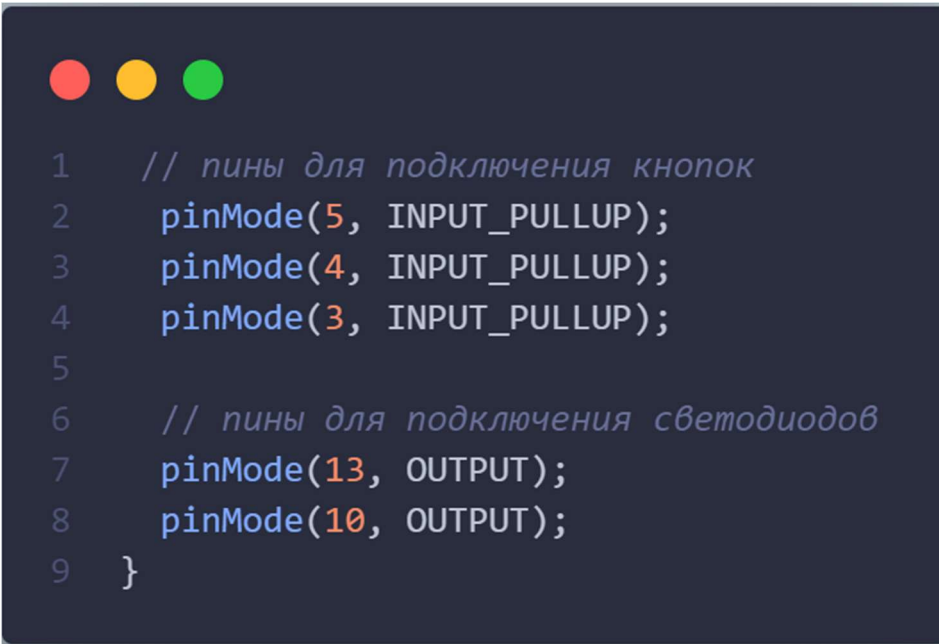
```

1  byte x;
2  byte w = 1;
3  String otvet = "";
4  String KOD = "";
5
6  // массив с вариантами кода
7  char* kod[6] = {"123", "132", "231", "213", "321", "312"};
8
9  // переменные для считывания нажатия на кнопки
10 boolean button1WasUp = true;
11 boolean button2WasUp = true;
12 boolean button3WasUp = true;
13
14 void setup() {
15     Serial.begin(9600);
16

```

Рисунок 1 Массив с вариантами кода

2. Пины для подключения кнопок и пины для подключения светодиодов



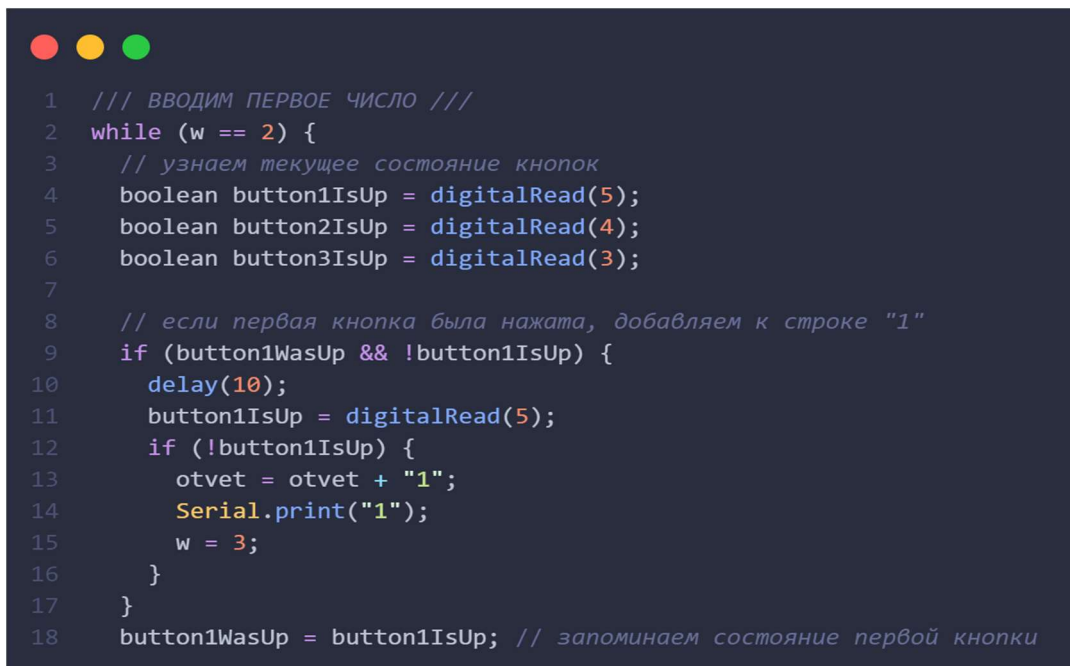
```

1  // пины для подключения кнопок
2  pinMode(5, INPUT_PULLUP);
3  pinMode(4, INPUT_PULLUP);
4  pinMode(3, INPUT_PULLUP);
5
6  // пины для подключения светодиодов
7  pinMode(13, OUTPUT);
8  pinMode(10, OUTPUT);
9  }

```

Рисунок 2 Пины для подключения кнопок

3. Вводим первое число и узнаем текущее состояние кнопок и если первая кнопка была нажата, добавляем к строке "1", запоминаем состояние и первой кнопки



```

1  /// ВВОДИМ ПЕРВОЕ ЧИСЛО ///
2  while (w == 2) {
3      // узнаем текущее состояние кнопок
4      boolean button1IsUp = digitalRead(5);
5      boolean button2IsUp = digitalRead(4);
6      boolean button3IsUp = digitalRead(3);
7
8      // если первая кнопка была нажата, добавляем к строке "1"
9      if (button1WasUp && !button1IsUp) {
10         delay(10);
11         button1IsUp = digitalRead(5);
12         if (!button1IsUp) {
13             otvet = otvet + "1";
14             Serial.print("1");
15             w = 3;
16         }
17     }
18     button1WasUp = button1IsUp; // запоминаем состояние первой кнопки

```

Рисунок 3 Вводим первое число

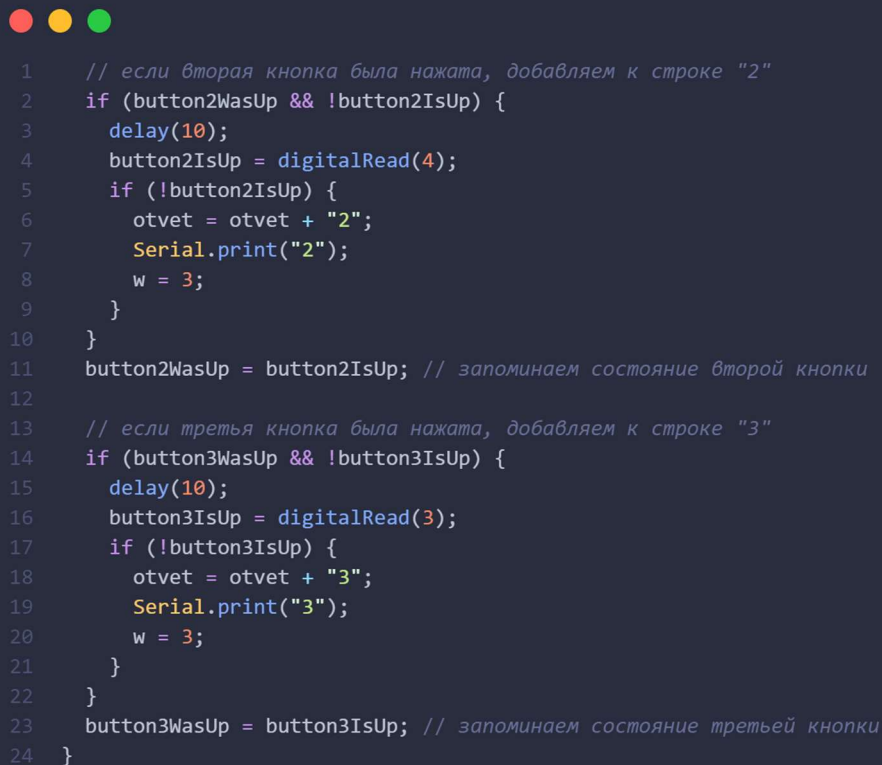
4 Функция генерации кода



```
1 void loop() {
2     /// ФУНКЦИЯ ГЕНЕРАЦИИ КОДА ///
3     while (w == 1) {
4         delay(500);
5         digitalWrite(13, HIGH);
6         digitalWrite(10, HIGH);
7         delay(100);
8         digitalWrite(13, LOW);
9         digitalWrite(10, LOW);
10        x = random(0, 5);
11        KOD = String(kod[x]);
12        Serial.print("KOD - ");
13        Serial.println(KOD);
14        Serial.println("");
15        Serial.print("OTVET - ");
16        w = 2;
17    }
18 }
```

Рисунок 4 Функция генрации кода

5. Если вторая кнопка была нажата, добавляем к строке "2", запоминаем состояние второй кнопки, если третья кнопка была нажата, добавляем к строке "3", запоминаем состояние третьей кнопки



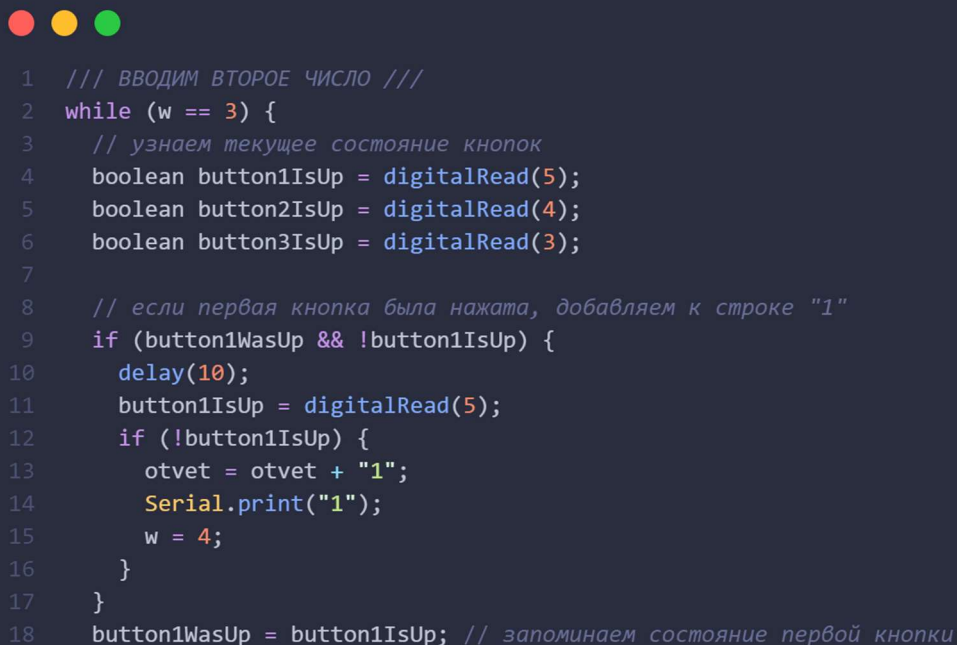
```

1  // если вторая кнопка была нажата, добавляем к строке "2"
2  if (button2WasUp && !button2IsUp) {
3      delay(10);
4      button2IsUp = digitalRead(4);
5      if (!button2IsUp) {
6          otvet = otvet + "2";
7          Serial.print("2");
8          w = 3;
9      }
10 }
11 button2WasUp = button2IsUp; // запоминаем состояние второй кнопки
12
13 // если третья кнопка была нажата, добавляем к строке "3"
14 if (button3WasUp && !button3IsUp) {
15     delay(10);
16     button3IsUp = digitalRead(3);
17     if (!button3IsUp) {
18         otvet = otvet + "3";
19         Serial.print("3");
20         w = 3;
21     }
22 }
23 button3WasUp = button3IsUp; // запоминаем состояние третьей кнопки
24 }

```

Рисунок 5 Если вторая кнопка была нажата, добавляем к строке "2"

6. Вводим второе число и узнаем текущее состояние кнопок, если первая кнопка была нажата, добавляем к строке "1"



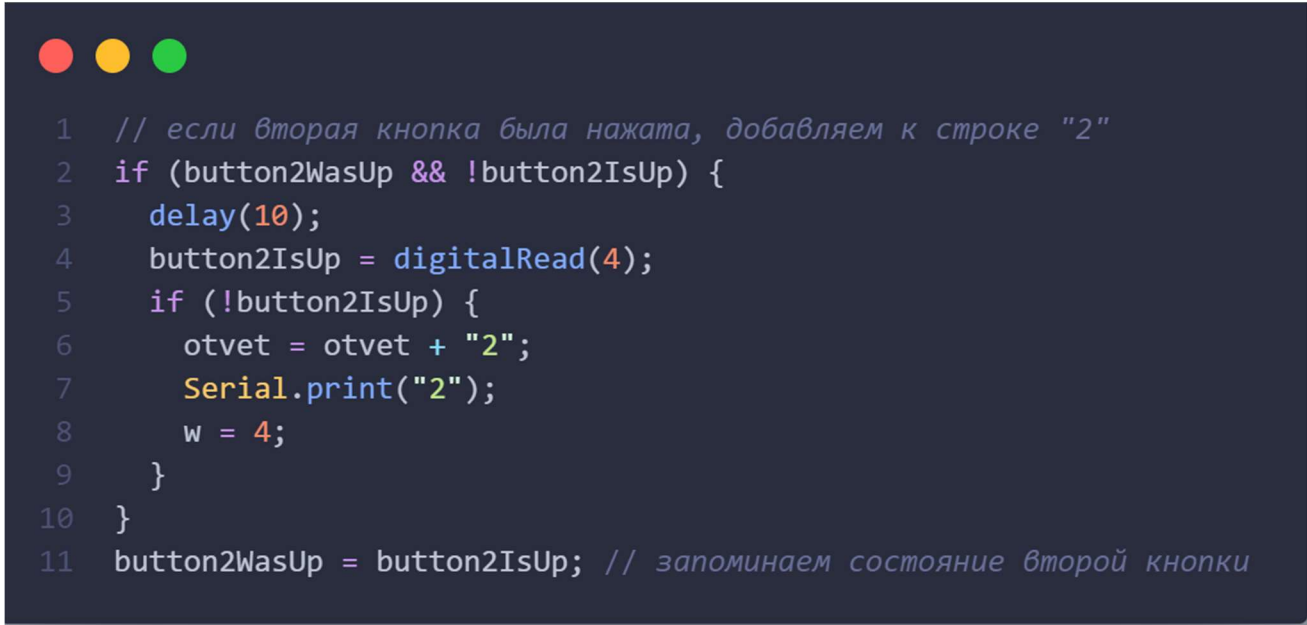
```

1  /// ВВОДИМ ВТОРОЕ ЧИСЛО ///
2  while (w == 3) {
3      // узнаем текущее состояние кнопок
4      boolean button1IsUp = digitalRead(5);
5      boolean button2IsUp = digitalRead(4);
6      boolean button3IsUp = digitalRead(3);
7
8      // если первая кнопка была нажата, добавляем к строке "1"
9      if (button1WasUp && !button1IsUp) {
10         delay(10);
11         button1IsUp = digitalRead(5);
12         if (!button1IsUp) {
13             otvet = otvet + "1";
14             Serial.print("1");
15             w = 4;
16         }
17     }
18     button1WasUp = button1IsUp; // запоминаем состояние первой кнопки

```

Рисунок 6 Вводим второе число

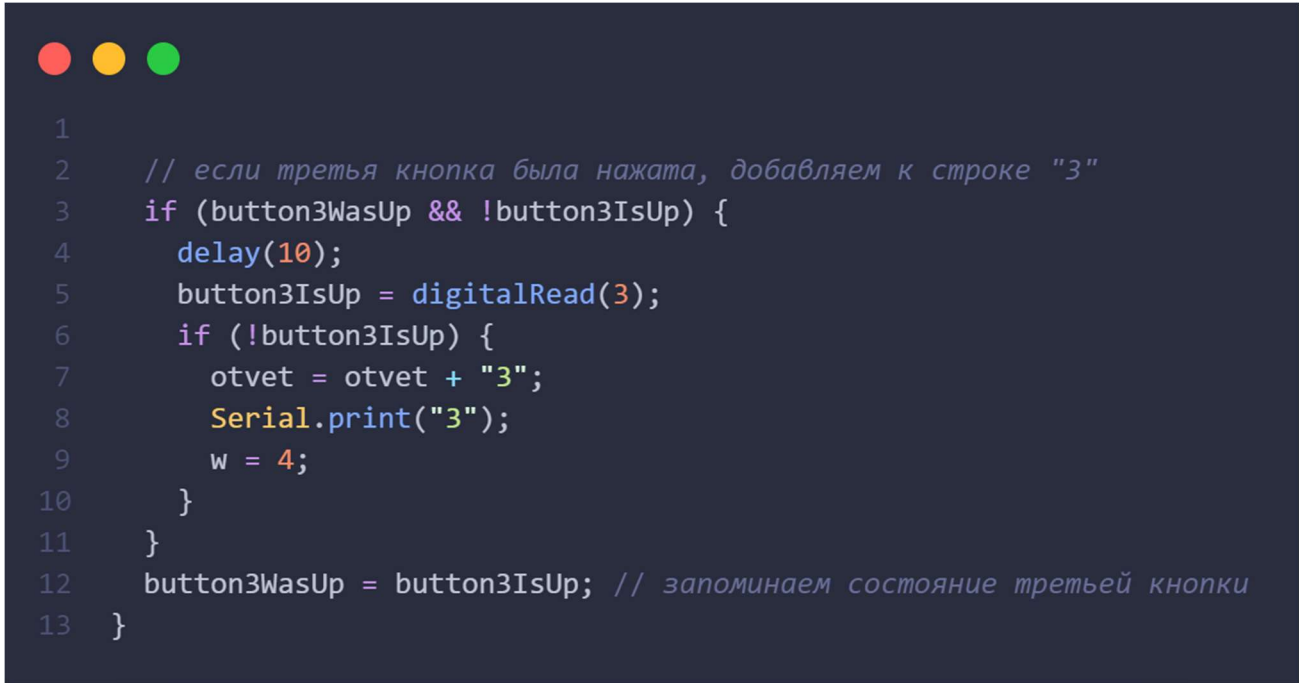
7. Если вторая кнопка была нажата, добавляем к строке "2", запоминаем состояние второй кнопки

A screenshot of a code editor with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The code is written in C++ and is numbered from 1 to 11. It implements logic for a second button: if the button was pressed and is currently up, it delays for 10ms, reads the button state, and if it's still up, it appends "2" to the output string, prints it, and sets a variable 'w' to 4. Finally, it updates 'button2WasUp' to the current state of 'button2IsUp'.

```
1 // если вторая кнопка была нажата, добавляем к строке "2"
2 if (button2WasUp && !button2IsUp) {
3     delay(10);
4     button2IsUp = digitalRead(4);
5     if (!button2IsUp) {
6         otvet = otvet + "2";
7         Serial.print("2");
8         w = 4;
9     }
10 }
11 button2WasUp = button2IsUp; // запоминаем состояние второй кнопки
```

Рисунок 7 Если вторая кнопка была нажата, добавляем к строке "2"

8. Если третья кнопка была нажата, добавляем к строке "3", запоминаем состояние третьей кнопки

A screenshot of a code editor with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The code is written in C++ and is numbered from 1 to 13. It implements logic for a third button: if the button was pressed and is currently up, it delays for 10ms, reads the button state, and if it's still up, it appends "3" to the output string, prints it, and sets a variable 'w' to 4. Finally, it updates 'button3WasUp' to the current state of 'button3IsUp'.

```
1
2 // если третья кнопка была нажата, добавляем к строке "3"
3 if (button3WasUp && !button3IsUp) {
4     delay(10);
5     button3IsUp = digitalRead(3);
6     if (!button3IsUp) {
7         otvet = otvet + "3";
8         Serial.print("3");
9         w = 4;
10    }
11 }
12 button3WasUp = button3IsUp; // запоминаем состояние третьей кнопки
13 }
```

Рисунок 8 Если третья кнопка была нажата, добавляем к строке "3"

9 Сравниваем наш ответ с кодом, стираем строку и генерируем новый код, стираем строку новая попытка ввода кода



```
1  /// СРАВНИВАЕМ НАШ ОТВЕТ С КОДОМ ///
```

```
2  if (KOD == otvet) {
```

```
3      Serial.println("");
```

```
4      Serial.println("DA!");
```

```
5      Serial.println("");
```

```
6      digitalWrite(10, HIGH);
```

```
7      delay(1000);
```

```
8      digitalWrite(10, LOW);
```

```
9      otvet = ""; // стираем строку
```

```
10     w = 1;      // генерируем новый код
```

```
11 } else {
```

```
12     Serial.println("");
```

```
13     Serial.println("HET!");
```

```
14     Serial.println("");
```

```
15     Serial.print("OTVET - ");
```

```
16     digitalWrite(13, HIGH);
```

```
17     delay(1000);
```

```
18     digitalWrite(13, LOW);
```

```
19     otvet = ""; // стираем строку
```

```
20     w = 2;      // новая попытка ввода кода
```

```
21 }
```

```
22 }
```

Рисунок 9 Сравниваем наш ответ с кодом

**Заключение.** Данную программу можно использовать в различных проектах, например, сделать будильник на Ардуино, где звуковой сигнал будет отключаться только при правильном вводе шифра. Или сделать кодовый замок, но в этом случае шифр не должен каждый раз генерироваться (иначе его сможет открыть любой человек), а шифр при этом можно сделать сложнее, состоящим из более, чем трех чисел