

Кнопочные ковбои

В этом эксперименте мы создаем игрушку на реакцию: кто быстрее нажмет кнопку по сигналу.

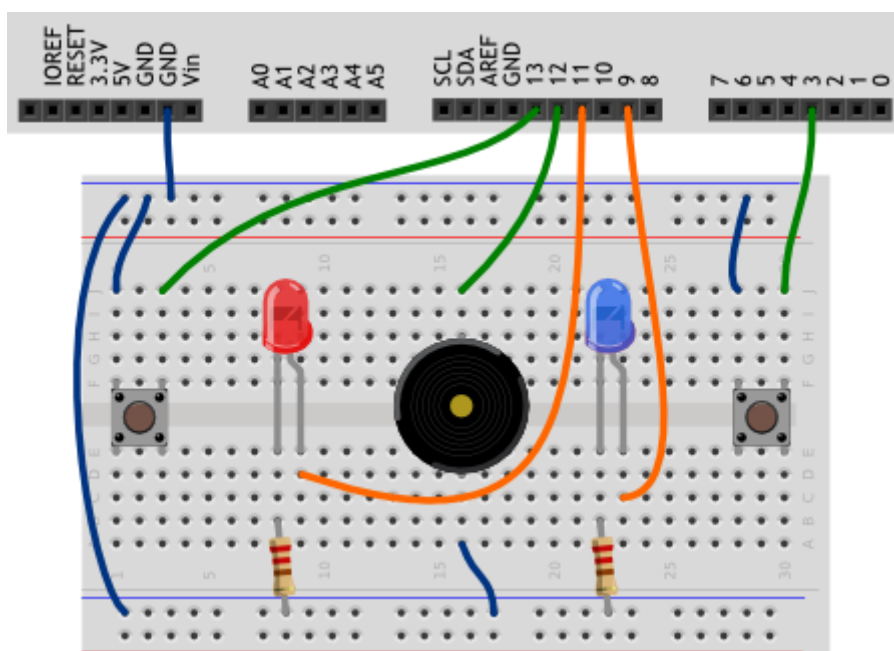
Задание 1. Ответить на вопросы

1. Как проверить, нажаты ли 2 кнопки одновременно?
2. Что такое массив?
3. Как обратиться к элементу массива?

Задание 2. Список деталей для эксперимента

1. 1 плата Arduino Uno
2. 1 беспаячная макетная плата
3. 2 тактовых кнопки
4. 2 резистора 220 Ом
5. 2 светодиода
6. 1 пьезопищалка
7. 10 проводов «папа-папа»

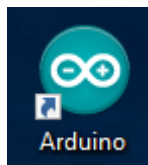
Схема на макетной плате:



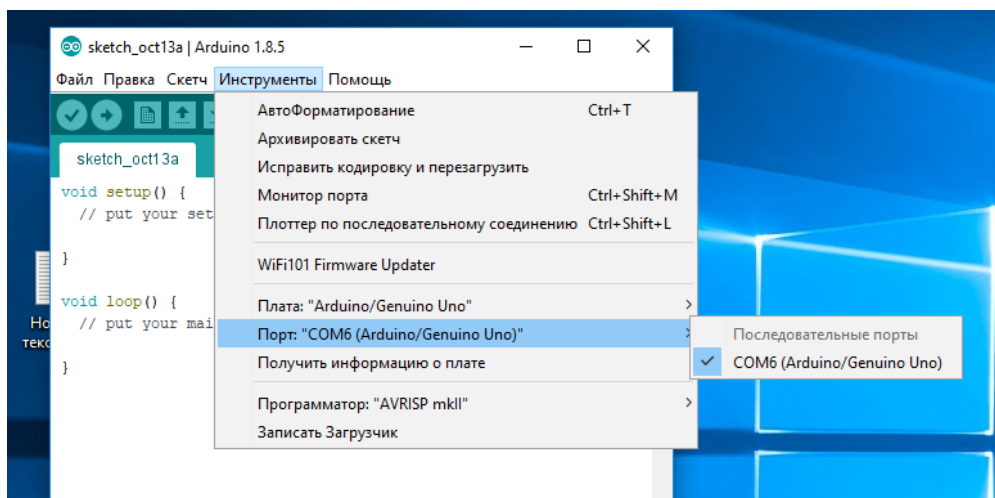
1. Зарисуйте принципиальную схему установки.

Задание 3. Программирование микроконтроллера

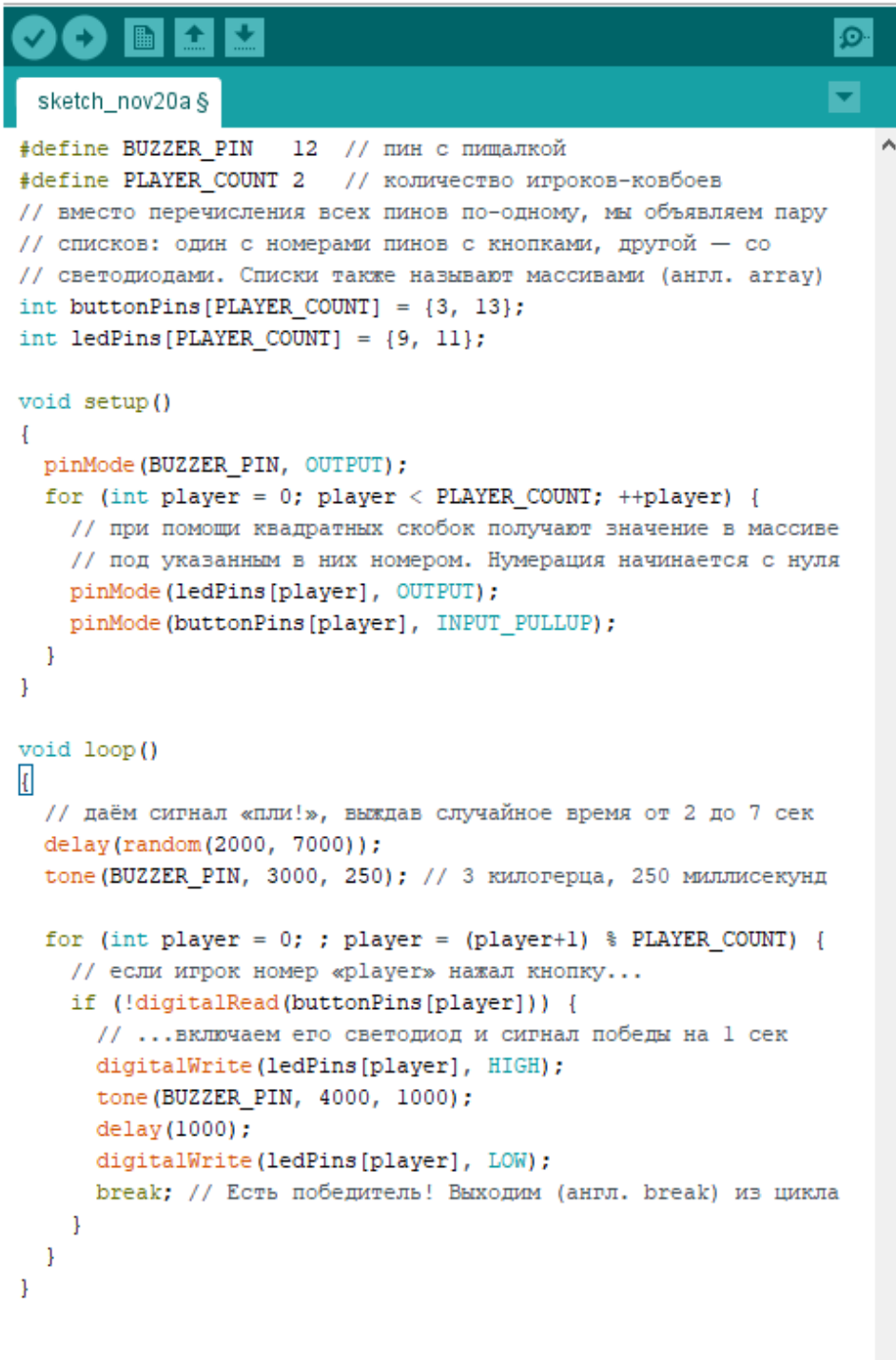
1. Запустите приложение



2. Убедитесь, что выбран нужный порт



3. Наберите в редакторе кода следующий код программы:



```

sketch_nov20a $
#define BUZZER_PIN 12 // пин с пищалкой
#define PLAYER_COUNT 2 // количество игроков-ковбоев
// вместо перечисления всех пинов по-одному, мы объявляем пару
// списков: один с номерами пинов с кнопками, другой — со
// светодиодами. Списки также называют массивами (англ. array)
int buttonPins[PLAYER_COUNT] = {3, 13};
int ledPins[PLAYER_COUNT] = {9, 11};

void setup()
{
  pinMode(BUZZER_PIN, OUTPUT);
  for (int player = 0; player < PLAYER_COUNT; ++player) {
    // при помощи квадратных скобок получают значение в массиве
    // под указанным в них номером. Нумерация начинается с нуля
    pinMode(ledPins[player], OUTPUT);
    pinMode(buttonPins[player], INPUT_PULLUP);
  }
}

void loop()
{
  // даём сигнал «пли!», выждав случайное время от 2 до 7 сек
  delay(random(2000, 7000));
  tone(BUZZER_PIN, 3000, 250); // 3 килогерца, 250 миллисекунд

  for (int player = 0; ; player = (player+1) % PLAYER_COUNT) {
    // если игрок номер «player» нажал кнопку...
    if (!digitalRead(buttonPins[player])) {
      // ...включаем его светодиод и сигнал победы на 1 сек
      digitalWrite(ledPins[player], HIGH);
      tone(BUZZER_PIN, 4000, 1000);
      delay(1000);
      digitalWrite(ledPins[player], LOW);
      break; // Есть победитель! Выходим (англ. break) из цикла
    }
  }
}

```

Пояснения к коду

- ✓ Массив состоит из элементов одного типа, в нашем случае int.
- ✓ Объявить массив можно следующими способами:

```
int firstArray[6]; // 6 целых чисел с неопределёнными начальными значениями
int pwmPins[] = {3, 5, 6, 9, 10, 11}; // 6 целых чисел, длина вычисляется
автоматом
```

`boolean buttonState[3] = {false, true, false};` // можно использовать элементы любого типа

✓ Когда мы объявляем массив с указанием количества его элементов `n`, это число всегда на 1 больше, чем номер последнего элемента (`n-1`), т.к. индекс первого элемента — 0.

✓ Считать или записать значение элемента массива можно, обратившись к нему по индексу, например `firstArray[2]` или `buttonState[counter]`, где `counter` — переменная, такая как счетчик цикла

✓ В переменных типа `long` можно хранить значения до 2 147 483 647. `unsigned int` в этом случае нам будет недостаточно, потому что 65 535 миллисекунд пройдут чуть больше чем за минуту!

✓ Функция `random(min, max)` возвращает целое псевдослучайное число в интервале `[min, max]`. Для драматичности каждая игра начинается с паузы случайной длины.

✓ Благодаря массивам в этом эксперименте мы настраиваем порты, считываем кнопки и включаем светодиоды в циклах со счетчиком, который используется как индекс элемента.

✓ Мы используем цикл `for` без условия его завершения, поэтому пока мы явно того не потребуем, цикл будет крутиться до бесконечности.

✓ Мы использовали выражение `player = (player+1) % PLAYER_COUNT` для счётчика цикла, чтобы не только увеличивать его на единицу каждый раз, но и обнулять при достижении последнего игрока.

✓ Инструкция `break` прекращает работу цикла и выполнение программы продолжается с инструкции после его конца.

Задание 4. Ответьте на следующие вопросы

1. Можно ли поместить в один массив элементы типа `boolean` и `int`?
2. Обязательно ли при объявлении массива заполнять его значениями?
3. Чем удобно использование массива?
4. Как обратиться к элементу массива, чтобы прочесть его значение?

5. Почему для хранения времени прошлого сигнала мы используем переменную типа `long`?
6. Чем отличаются инструкции `continue` и `break`?

Задание 5. Самостоятельно измените существующую программу и схему

1. Сделайте напряженный вариант игры: пусть интервал между сигналами будет в диапазоне от 10 до 15 секунд.
2. В игре есть лазейка: кнопку можно зажать до сигнала «пли!» и таким образом сразу же выиграть. Дополните программу так, чтобы так выиграть было нельзя.
3. Добавьте в игру еще двух ковбоев!