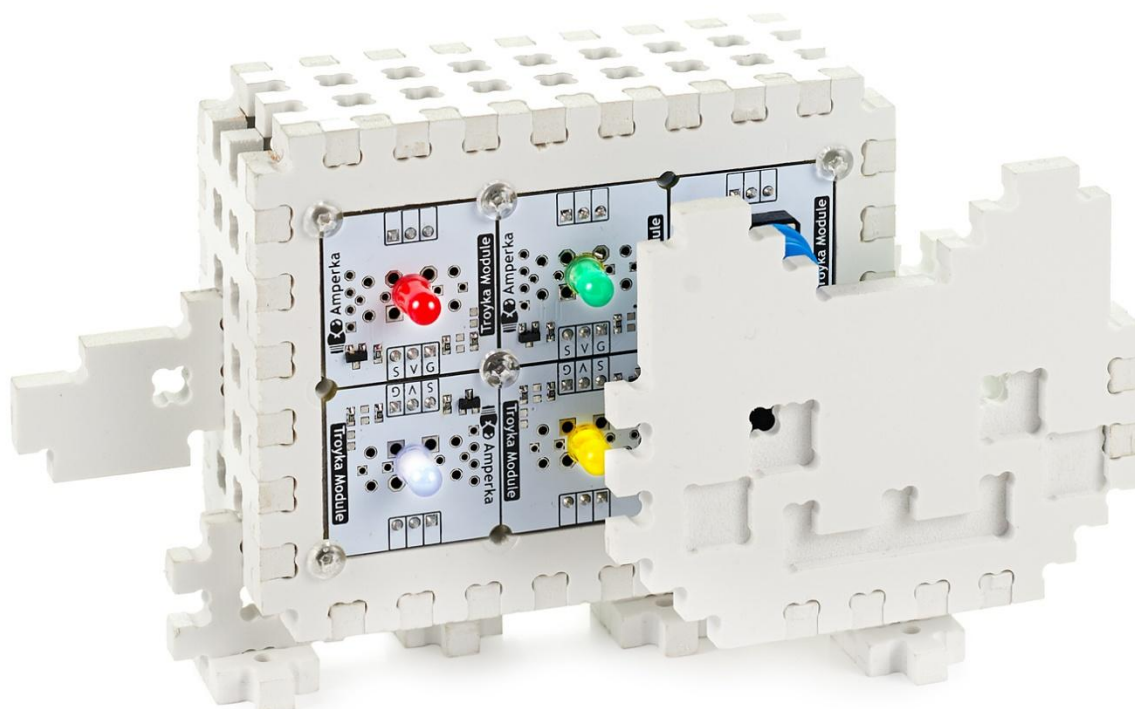
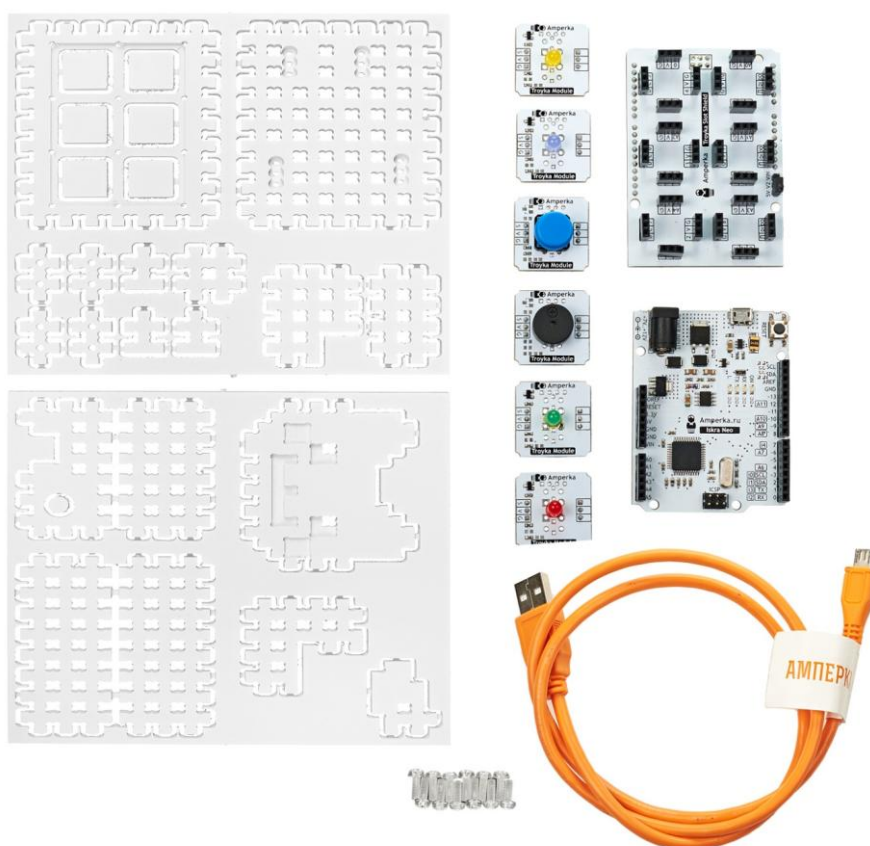


NYAN!



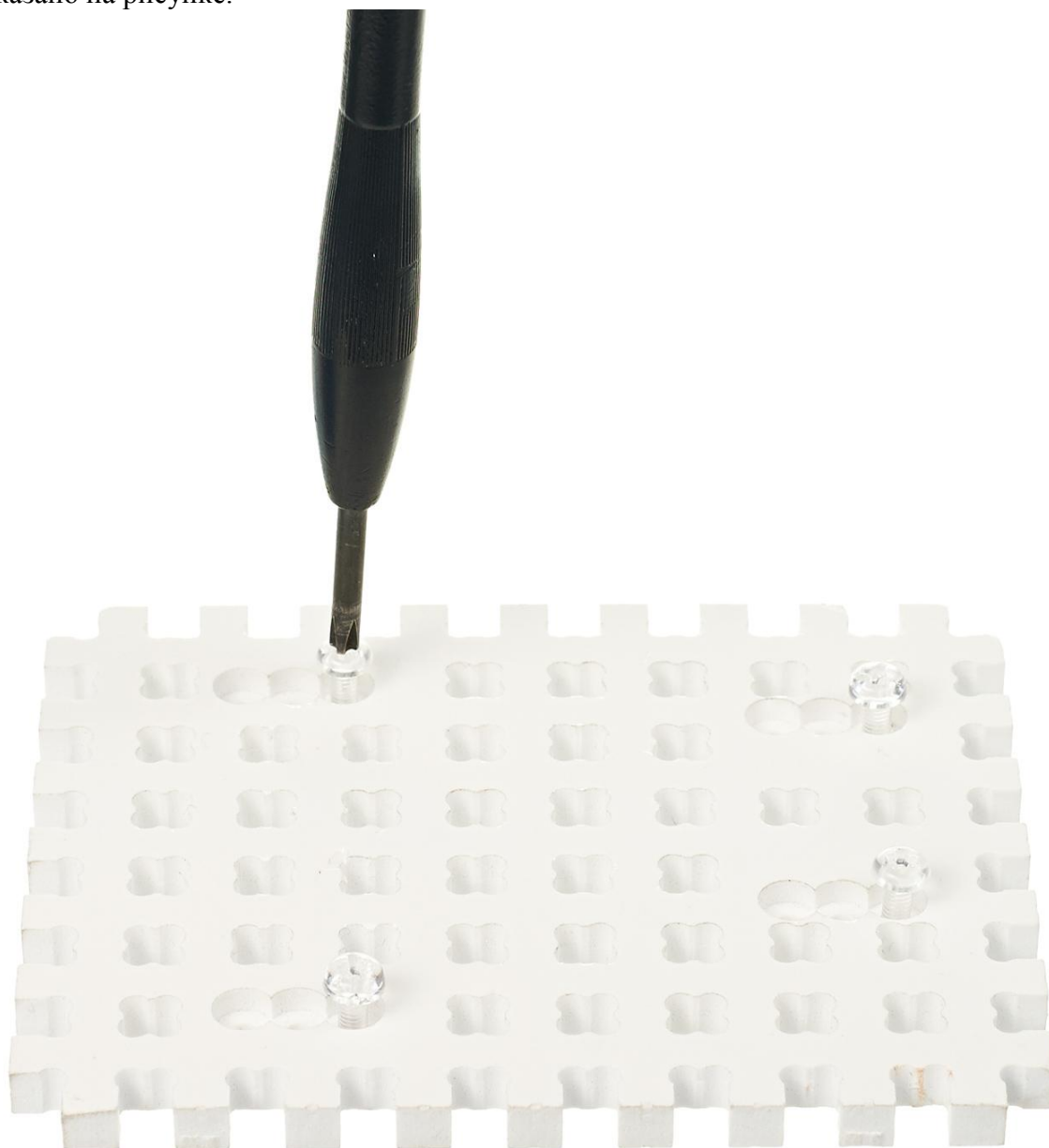
Что входит в набор?



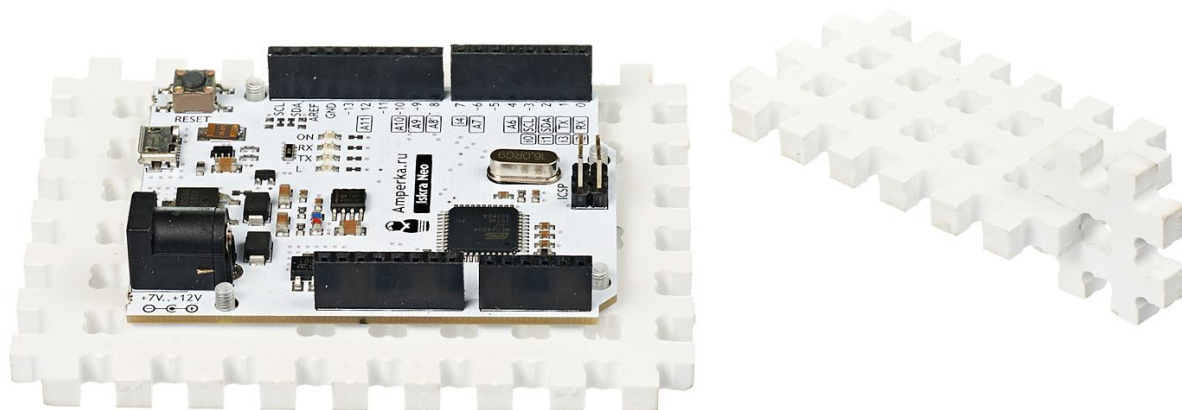
1. [Iskra Neo](#)
2. [Тройка Slot Shield](#)
3. [Светодиоды 5 мм \(Тройка-модуль\)](#) красный, жёлтый, зелёный и синий
4. [Кабель USB \(A — Micro USB\)](#)
5. 12× [Винт М3](#)
6. [Набор деталей Slot Box \(#Структор\)](#)

Как собрать

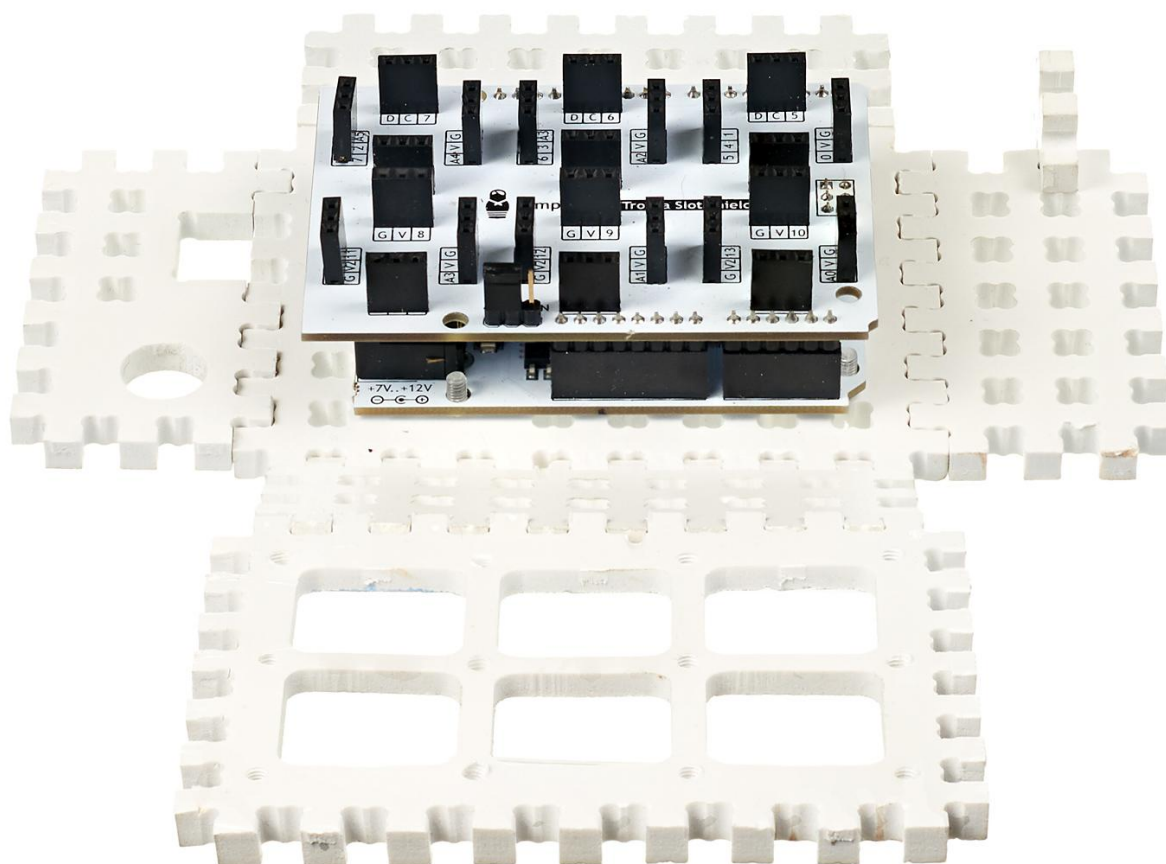
Шаг первый. Закрепите микроконтроллер на плашке #структора. Вкрутите винты М3 как показано на рисунке.



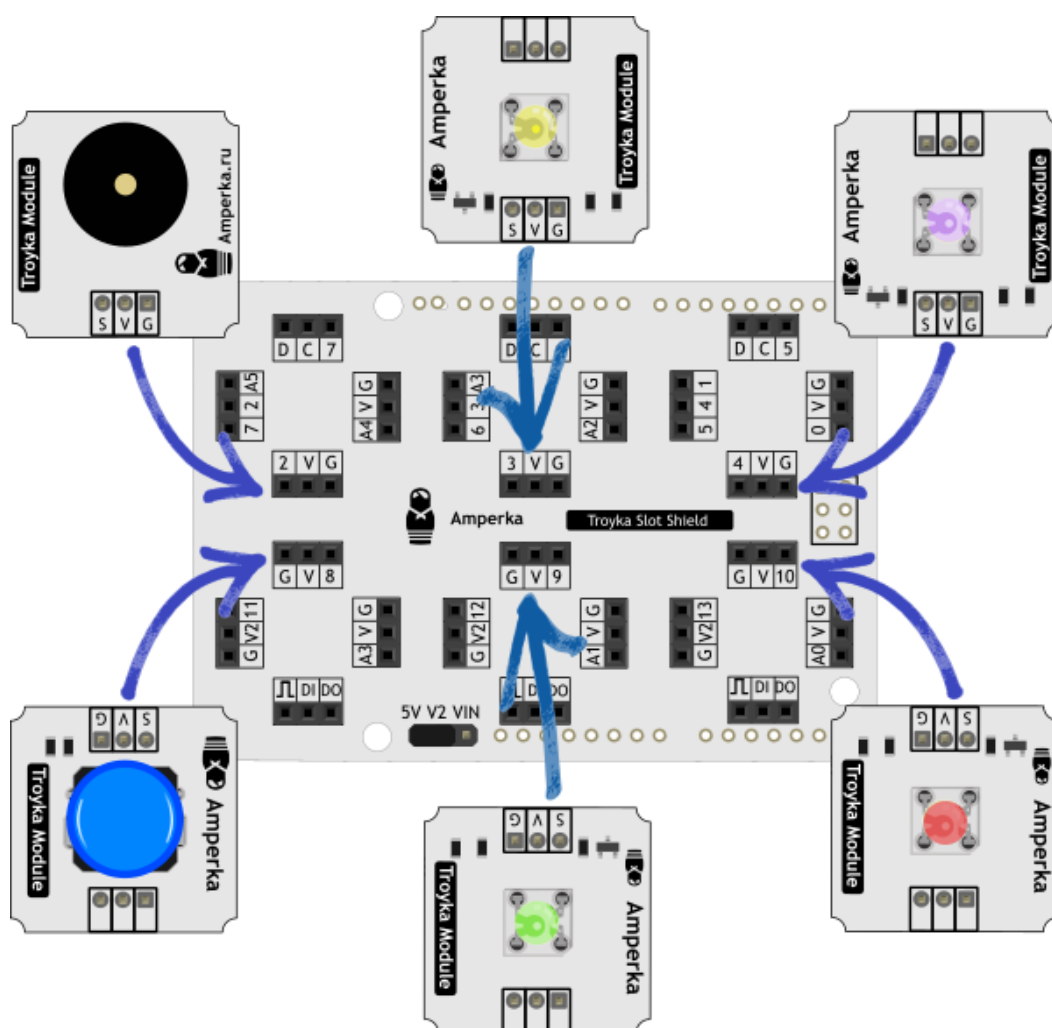
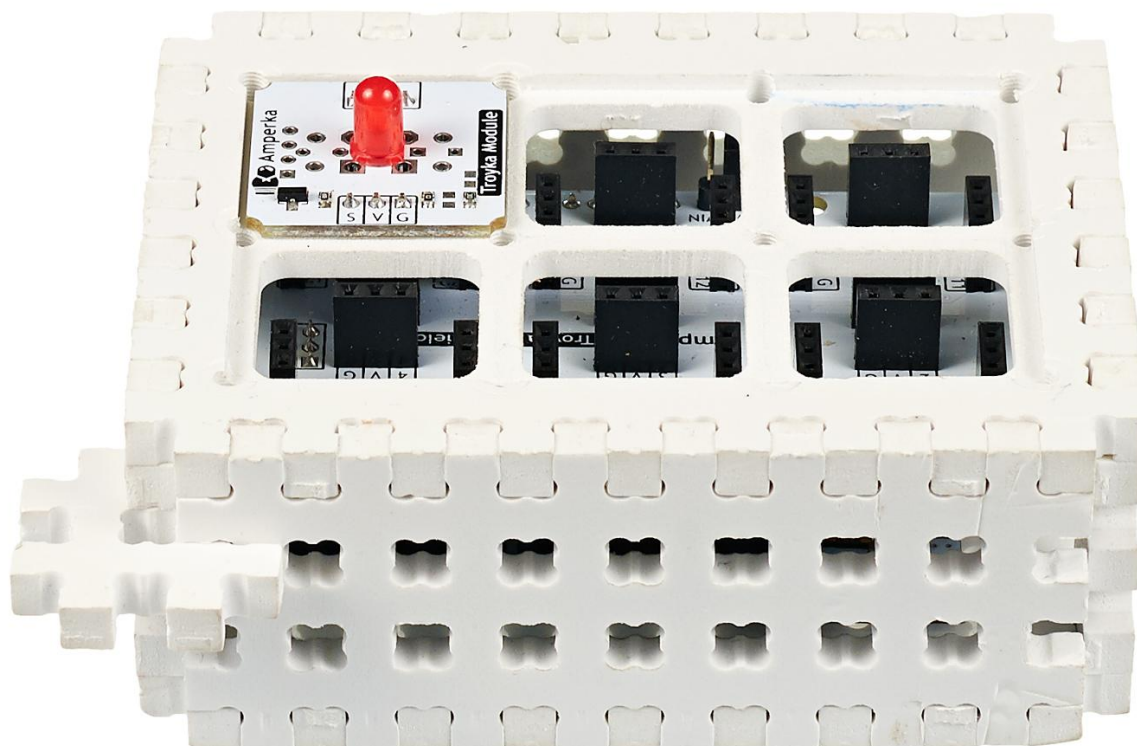
Шаг второй. Переверните плашку и закрепите микроконтроллер.



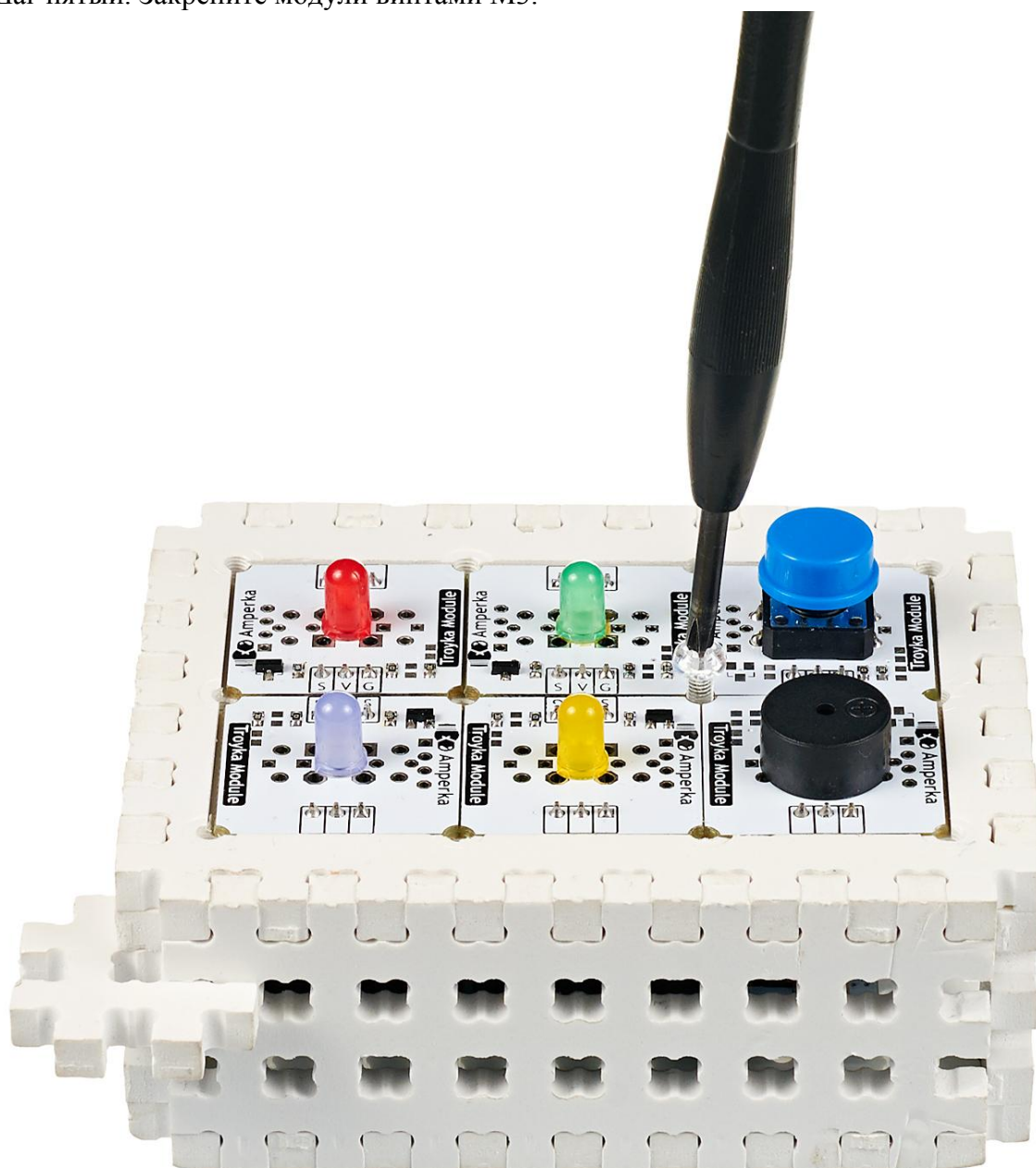
Шаг третий. Соберите боковые стенки корпуса, затем закрепите верхнюю панель.



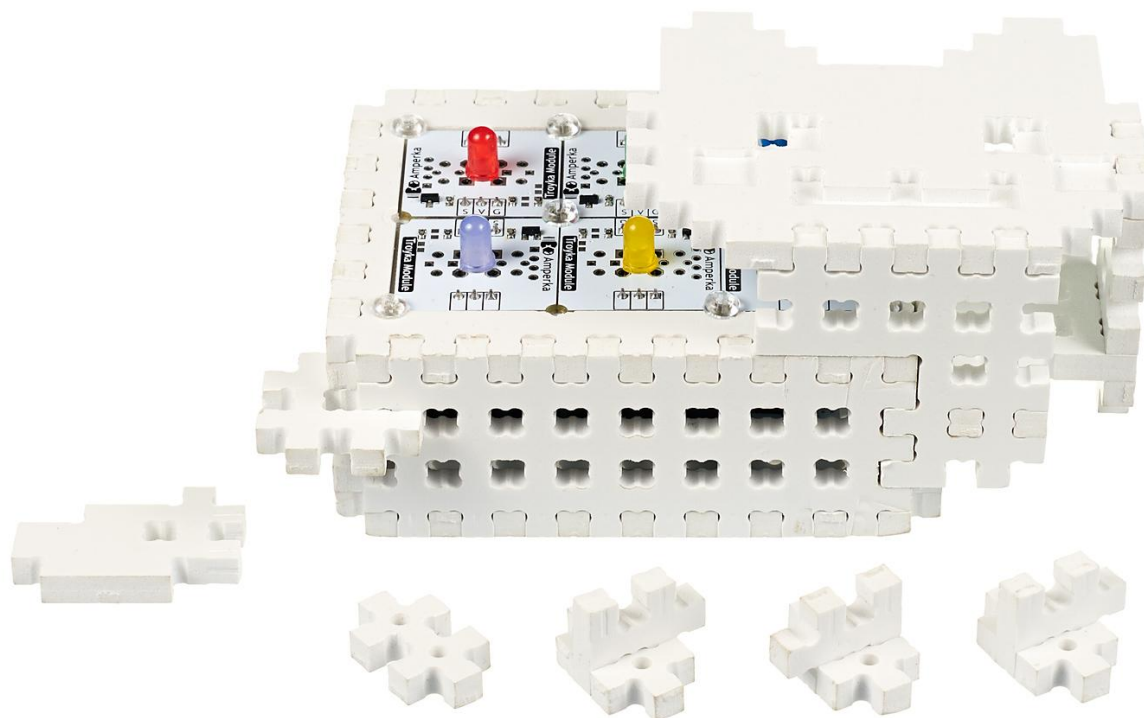
Шаг четвёртый. Вставьте светодиоды, баззер и кнопку. Порядок установки приведён на рисунке.



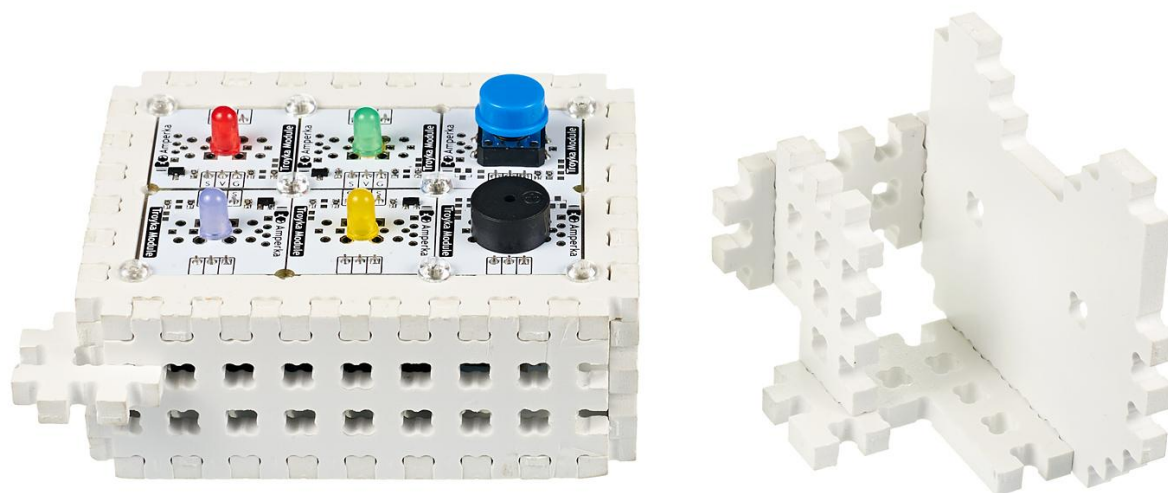
Шаг пятый. Закрепите модули винтами М3.



Шаг шестой. Соберите голову котика и закрепите на корпусе.



Шаг седьмой. Добавьте лапы.



Ваш котик готов.



Программируй, Ня!

1. Скачайте и установите [Arduino IDE](#);
2. Подключите кота к компьютеру через кабель micro-USB. В системе появится новый COM-порт;
3. Настройте Arduino IDE, выбрав в меню Инструменты Плата Arduino Leonardo ;
4. Укажите нужный COM-порт в меню Инструменты Порт COMXX ;
5. Скачайте и откройте в Arduino IDE скетч [NyanCat.ino](#) из раздела [Нян Код](#);
6. Нажмите на кнопку Скетч Загрузка Ctrl+U или просто ;
7. Ня!

Нян Код

[NyanCat.ino](#)

```
/*Nyan Cat
  with Bass
  Uses Arduino tone library pitches.h
  [http://arduino.cc/en/Tutorial/tone ]
  by electricmango
  https://electricmango.github.io

  Thanks to Sean for posting the sheet music on
  http://junket.tumblr.com/post/4776023022/heres-the-nyan-cat-sheet-
  music

  Version 1.0.3
  -----
  Nyan Cat with Bass is licensed under the
```

Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0)

You are free to:

Share – copy and redistribute the material in any medium or format

Adapt – remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution – You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

ShareAlike – If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

No additional restrictions – You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

The full license is available at <https://creativecommons.org/licenses/by-sa/4.0/legalcode>

Copyright (c) 2012 ~ 2015 electricmango

Edit by Amperka.ru in 2016

Nyan_Cat.ino

*/

//Ноты

```
#define REST 0
#define NOTE_B0 31
#define NOTE_C1 33
#define NOTE_CS1 35
#define NOTE_D1 37
#define NOTE_DS1 39
#define NOTE_E1 41
#define NOTE_F1 44
#define NOTE_FS1 46
#define NOTE_G1 49
#define NOTE_GS1 52
#define NOTE_A1 55
#define NOTE_AS1 58
#define NOTE_B1 62
#define NOTE_C2 65
#define NOTE_CS2 69
#define NOTE_D2 73
#define NOTE_DS2 78
#define NOTE_E2 82
#define NOTE_F2 87
#define NOTE_FS2 93
#define NOTE_G2 98
#define NOTE_GS2 104
#define NOTE_A2 110
#define NOTE_AS2 117
#define NOTE_B2 123
#define NOTE_C3 131
#define NOTE_CS3 139
```

```
#define NOTE_D3 147
#define NOTE_DS3 156
#define NOTE_E3 165
#define NOTE_F3 175
#define NOTE_FS3 185
#define NOTE_G3 196
#define NOTE_GS3 208
#define NOTE_A3 220
#define NOTE_AS3 233
#define NOTE_B3 247
#define NOTE_C4 262
#define NOTE_CS4 277
#define NOTE_D4 294
#define NOTE_DS4 311
#define NOTE_E4 330
#define NOTE_F4 349
#define NOTE_FS4 370
#define NOTE_G4 392
#define NOTE_GS4 415
#define NOTE_A4 440
#define NOTE_AS4 466
#define NOTE_B4 494
#define NOTE_C5 523
#define NOTE_CS5 554
#define NOTE_D5 587
#define NOTE_DS5 622
#define NOTE_E5 659
#define NOTE_F5 698
#define NOTE_FS5 740
#define NOTE_G5 784
#define NOTE_GS5 831
#define NOTE_A5 880
#define NOTE_AS5 932
#define NOTE_B5 988
#define NOTE_C6 1047
#define NOTE_CS6 1109
#define NOTE_D6 1175
#define NOTE_DS6 1245
#define NOTE_E6 1319
#define NOTE_F6 1397
#define NOTE_FS6 1480
#define NOTE_G6 1568
#define NOTE_GS6 1661
#define NOTE_A6 1760
#define NOTE_AS6 1865
#define NOTE_B6 1976
#define NOTE_C7 2093
#define NOTE_CS7 2217
#define NOTE_D7 2349
#define NOTE_DS7 2489
#define NOTE_E7 2637
#define NOTE_F7 2794
#define NOTE_FS7 2960
#define NOTE_G7 3136
#define NOTE_GS7 3322
#define NOTE_A7 3520
#define NOTE_AS7 3729
#define NOTE_B7 3951
#define NOTE_C8 4186
#define NOTE_CS8 4435
#define NOTE_D8 4699
#define NOTE_DS8 4978
#define LED_R 10
```

```

#define LED_Y 3
#define LED_G 9
#define LED_B 4
#define BUZZER 2
#define BUTTON 8

// Массив с нотами мелодии

int melody[] = {
    NOTE_DS5, NOTE_E5, NOTE_FS5, 0, NOTE_B5, NOTE_E5, NOTE_DS5, NOTE_E5,
    NOTE_FS5, NOTE_B5, NOTE_DS6, NOTE_E6, NOTE_DS6, NOTE_AS5, NOTE_B5, 0,
    NOTE_FS5, 0, NOTE_DS5, NOTE_E5, NOTE_FS5, 0, NOTE_B5, NOTE_CS6,
    NOTE_AS5, NOTE_B5, NOTE_CS6, NOTE_E6, NOTE_DS6, NOTE_E6, NOTE_CS6,
    NOTE_FS4, NOTE_GS4, NOTE_D4, NOTE_DS4, NOTE_FS2, NOTE_CS4, NOTE_D4,
    NOTE_CS4, NOTE_B3, NOTE_B3, NOTE_CS4,
    NOTE_D4, NOTE_D4, NOTE_CS4, NOTE_B3, NOTE_CS4, NOTE_DS4, NOTE_FS4,
    NOTE_GS4, NOTE_DS4, NOTE_FS4, NOTE_CS4, NOTE_DS4, NOTE_B3, NOTE_CS4,
    NOTE_B3,
    NOTE_DS4, NOTE_FS4, NOTE_GS4, NOTE_DS4, NOTE_FS4, NOTE_CS4, NOTE_DS4,
    NOTE_B3, NOTE_D4, NOTE_DS4, NOTE_D4, NOTE_CS4, NOTE_B3, NOTE_CS4,
    NOTE_D4, NOTE_B3, NOTE_CS4, NOTE_DS4, NOTE_FS4, NOTE_CS4, NOTE_D4,
    NOTE_CS4, NOTE_B3, NOTE_CS4, NOTE_B3, NOTE_CS4,
    NOTE_FS4, NOTE_GS4, NOTE_D4, NOTE_DS4, NOTE_FS2, NOTE_CS4, NOTE_D4,
    NOTE_CS4, NOTE_B3, NOTE_B3, NOTE_CS4,
    NOTE_D4, NOTE_D4, NOTE_CS4, NOTE_B3, NOTE_CS4, NOTE_DS4, NOTE_FS4,
    NOTE_GS4, NOTE_DS4, NOTE_FS4, NOTE_CS4, NOTE_DS4, NOTE_B3, NOTE_CS4,
    NOTE_B3,
    NOTE_DS4, NOTE_FS4, NOTE_GS4, NOTE_DS4, NOTE_FS4, NOTE_CS4, NOTE_DS4,
    NOTE_B3, NOTE_D4, NOTE_DS4, NOTE_D4, NOTE_CS4, NOTE_B3, NOTE_CS4,
    NOTE_D4, NOTE_B3, NOTE_CS4, NOTE_DS4, NOTE_FS4, NOTE_CS4, NOTE_D4,
    NOTE_CS4, NOTE_B3, NOTE_CS4, NOTE_B3, NOTE_B3,
    NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3,
    NOTE_CS4, NOTE_DS4, NOTE_B3, NOTE_E4, NOTE_DS4, NOTE_E4, NOTE_FS4,
    NOTE_B3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3, NOTE_FS3, NOTE_E4,
    NOTE_DS4, NOTE_CS4, NOTE_B3, NOTE_E3, NOTE_DS3, NOTE_E3, NOTE_FS3,
    NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3,
    NOTE_B3, NOTE_CS4, NOTE_DS4, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_FS3,
    NOTE_B3, NOTE_B3, NOTE_AS3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3,
    NOTE_E4, NOTE_DS4, NOTE_E4, NOTE_FS4, NOTE_B3, NOTE_AS3,
    NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3,
    NOTE_CS4, NOTE_DS4, NOTE_B3, NOTE_E4, NOTE_DS4, NOTE_E4, NOTE_FS4,
    NOTE_B3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3, NOTE_FS3, NOTE_E4,
    NOTE_DS4, NOTE_CS4, NOTE_B3, NOTE_E3, NOTE_DS3, NOTE_E3, NOTE_FS3,
    NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3,
    NOTE_B3, NOTE_CS4, NOTE_DS4, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_FS3,
    NOTE_B3, NOTE_B3, NOTE_AS3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3,
    NOTE_E4, NOTE_DS4, NOTE_E4, NOTE_FS4, NOTE_B3, NOTE_CS4,
    NOTE_FS4, NOTE_GS4, NOTE_D4, NOTE_DS4, NOTE_FS2, NOTE_CS4, NOTE_D4,
    NOTE_CS4, NOTE_B3, NOTE_B3, NOTE_CS4,
    NOTE_D4, NOTE_D4, NOTE_CS4, NOTE_B3, NOTE_CS4, NOTE_DS4, NOTE_FS4,
    NOTE_GS4, NOTE_DS4, NOTE_FS4, NOTE_CS4, NOTE_DS4, NOTE_B3, NOTE_CS4,
    NOTE_B3,
    NOTE_DS4, NOTE_FS4, NOTE_GS4, NOTE_DS4, NOTE_FS4, NOTE_CS4, NOTE_DS4,
    NOTE_B3, NOTE_D4, NOTE_DS4, NOTE_D4, NOTE_CS4, NOTE_B3, NOTE_CS4,
    NOTE_D4, NOTE_B3, NOTE_CS4, NOTE_DS4, NOTE_FS4, NOTE_CS4, NOTE_D4,
    NOTE_CS4, NOTE_B3, NOTE_CS4, NOTE_B3, NOTE_CS4,
    NOTE_FS4, NOTE_GS4, NOTE_D4, NOTE_DS4, NOTE_FS2, NOTE_CS4, NOTE_D4,
    NOTE_CS4, NOTE_B3, NOTE_B3, NOTE_CS4,
    NOTE_D4, NOTE_D4, NOTE_CS4, NOTE_B3, NOTE_CS4, NOTE_DS4, NOTE_FS4,
    NOTE_GS4, NOTE_DS4, NOTE_FS4, NOTE_CS4, NOTE_DS4, NOTE_B3, NOTE_CS4,
    NOTE_B3,
    NOTE_DS4, NOTE_FS4, NOTE_GS4, NOTE_DS4, NOTE_FS4, NOTE_CS4, NOTE_DS4,
    NOTE_B3, NOTE_D4, NOTE_DS4, NOTE_D4, NOTE_CS4, NOTE_B3, NOTE_CS4,

```

```

    NOTE_D4, NOTE_B3, NOTE_CS4, NOTE_DS4, NOTE_FS4, NOTE_CS4, NOTE_D4,
    NOTE_CS4, NOTE_B3, NOTE_CS4, NOTE_B3, NOTE_B3,
    NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3,
    NOTE_CS4, NOTE_DS4, NOTE_B3, NOTE_E4, NOTE_DS4, NOTE_E4, NOTE_FS4,
    NOTE_B3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3, NOTE_FS3, NOTE_E4,
    NOTE_DS4, NOTE_CS4, NOTE_B3, NOTE_E3, NOTE_DS3, NOTE_E3, NOTE_FS3,
    NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3,
    NOTE_B3, NOTE_CS4, NOTE_DS4, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_FS3,
    NOTE_B3, NOTE_B3, NOTE_AS3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3,
    NOTE_E4, NOTE_DS4, NOTE_E4, NOTE_FS4, NOTE_B3, NOTE_AS3,
    NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3,
    NOTE_CS4, NOTE_DS4, NOTE_B3, NOTE_E4, NOTE_DS4, NOTE_E4, NOTE_FS4,
    NOTE_B3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3, NOTE_FS3, NOTE_E4,
    NOTE_DS4, NOTE_CS4, NOTE_B3, NOTE_E3, NOTE_DS3, NOTE_E3, NOTE_FS3,
    NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3,
    NOTE_B3, NOTE_CS4, NOTE_DS4, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_FS3,
    NOTE_B3, NOTE_B3, NOTE_AS3, NOTE_B3, NOTE_FS3, NOTE_GS3, NOTE_B3,
    NOTE_E4, NOTE_DS4, NOTE_E4, NOTE_FS4, NOTE_B3, NOTE_CS4,
};

// Массив длительности нот в мелодии: 4 = четвертная нота,
// 8 = восьмая нота, и так далее...
int noteDurations[] = {
    16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    16, 16, 16, 16, 16, 16, 8, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 8, 16, 16, 16, 16, 16, 16, 16, 8, 8, 8,
    8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 8, 8, 8,
    8, 8, 16, 16, 16, 16, 16, 16, 16, 8, 8, 8,
    8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 8, 8, 8,
    8, 16, 16, 8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 16, 16, 8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 8, 8,
    8, 16, 16, 8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 8, 8,
    8, 16, 16, 8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 16, 16, 8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,
    8, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 8,
};

void setup() {
    //iterate over the notes of the melody:
    pinMode(LED_R, OUTPUT);
    pinMode(LED_Y, OUTPUT);
}

```

```

    pinMode(LED_G, OUTPUT);
    pinMode(LED_B, OUTPUT);
}

void loop() {
    // Если была нажата кнопка, играем мелодию
    if (checkClick())
        nya();

    // Гасим светодиоды
    digitalWrite(LED_R, LOW);
    digitalWrite(LED_G, LOW);
    digitalWrite(LED_Y, LOW);
    digitalWrite(LED_B, LOW);
}

void nya()
{
    // Вычисляем количество нот
    int noteCount = sizeof(melody) / sizeof(int);

    // Переменная для хранения состояния светодиодов
    bool toggleLeds = true;

    // Играем мелодию
    for (int thisNote = 0; thisNote < noteCount; thisNote++) {

        // Если была нажата клавиша - останавливаем мелодию
        if (checkClick())
            return;

        // Каждую новую ноту меняем состояние светодиодов
        digitalWrite(LED_R, toggleLeds);
        digitalWrite(LED_Y, toggleLeds);
        digitalWrite(LED_G, !toggleLeds);
        digitalWrite(LED_B, !toggleLeds);
        toggleLeds = !toggleLeds;

        noTone(BUZZER); //останавливаем воспроизведение предыдущей ноты

        /*
        Вычислим длительность ноты. Разделим секунду на длительность ноты
из массива
        Четвертная нота = 1000/4, восьмая нота 1000/8
        Скорость можно увеличить, увеличив числитель. Больше подошло 1200
        */

        int noteDuration = 1200 / noteDurations[thisNote];

        // Можно увеличить высоту мелодии, умножая ноты на 2, 4, 8
        int octaveMul = 2;

        // воспроизведём ноту
        tone(BUZZER, octaveMul * melody[thisNote], noteDuration);

        /*
        Чтобы отделить ноты друг от друга, добавим небольшую паузу между
ними
        30% от длины ноты звучат неплохо
        */
        int pauseBetweenNotes = noteDuration * 1.30;
        delay(pauseBetweenNotes);
    }
}

```

```

    }
}
// Функция для проверки клика кнопки
bool checkClick()
{
    // Переменная для хранения предыдущего состояния кнопки
    static bool buttonWasUp = true;
    bool result = false;

    // Читаем текущее состояние кнопки
    bool buttonIsUp = digitalRead(BUTTON);

    // Если кнопка была не нажата, а теперь нажата
    // значит был клик
    if (buttonWasUp && !buttonIsUp) {
        delay(50);
        result = true;
    }
    buttonWasUp = buttonIsUp;
    return result;
}

```

Таймер «Помидор»

[Помидорная техника](#) используется для повышения собственной продуктивности. Вы просто ставите себе таймер на 25 минут и не отвлекаетесь. А ещё с её помощью можно приготовить яичницу.

[pomodoroTimer.ino](#)

```

// Время таймера в минутах
#define POMODORO_TIME 25

// Используемые пины
#define LED_R 10
#define LED_Y 3
#define LED_G 9
#define LED_B 4
#define BUZZER 2
#define BUTTON 8

// Количество светодиодов
#define LED_COUNT 4

// Массив пинов со светодиодами
int leds[] = {LED_R,
              LED_G,
              LED_Y,
              LED_B
            };

// Переменная для хранения времени таймера в миллисекундах
unsigned long pomodoroTimeInMillis;

// Переменная для хранения времени переключения светодиодов
unsigned long ledSwitchTime;

// Переменная для хранения времени начала работы таймера
unsigned long lastTime = 0;

// Состояние таймера – выключен
bool pomodoroState = false;

void setup() {

```

```

// Рассчитаем время таймера в миллисекундах
pomodoroTimeInMillis = POMODORO_TIME * 60000;

// Рассчитаем время переключения светодиодов
ledSwitchTime = pomodoroTimeInMillis / LED_COUNT;

// Переключим пины светодиодов в состояние выхода
for (int i = 0; i < LED_COUNT; ++i) {
    pinMode(leds[i], OUTPUT);
}
}

void loop() {
    // Если был клик кнопки
    if (checkClick()) {

        // Запомним текущее время
        lastTime = millis();

        // Переключим состояние таймера вкл/выкл
        pomodoroState = !pomodoroState;

        // Выключим все светодиоды
        for (int i = 0; i < LED_COUNT; ++i) {
            digitalWrite(leds[i], LOW);
        }

        // Если таймер включён
        if (pomodoroState) {

            // Рассчитаем прошедшее время
            unsigned long elapsedTime = millis() - lastTime;

            // Если прошедшее время меньше времени срабатывания таймера
            if (elapsedTime < pomodoroTimeInMillis) {

                // Рассчитаем, какой из светодиодов должен гореть
                unsigned long currentLed = elapsedTime / ledSwitchTime;

                // И зажжём этот светодиод
                digitalWrite(leds[currentLed], HIGH);

                // Если прошедшее время таймера больше времени срабатывания –
                звеним
            } else {
                finish();
            }
        }
    }

    // Функция для проверки клика кнопки
    bool checkClick()
    {
        // Переменная для хранения предыдущего состояния кнопки
        static bool buttonWasUp = true;
        bool result = false;

        // Читаем текущее состояние кнопки
        bool buttonIsUp = digitalRead(BUTTON);

        // Если кнопка была не нажата, а теперь нажата
        // значит был клик
    }
}

```

```

    if (buttonWasUp && !buttonIsUp) {
        delay(100);
        result = true;
    }
    buttonWasUp = buttonIsUp;
    return result;
}

void finish() {
    // Переменная для хранения переключателя светодиодов
    bool toggle = true;

    // Попиликаем 10 раз...
    for (int i = 0; i < 10; i++) {
        tone(BUZZER, 500 + toggle * 500);

        // ... поочерёдно моргая парами светодиодов
        for (int i = 0; i < LED_COUNT; ++i) {
            digitalWrite(leds[i], toggle);
            toggle = !toggle;
        }
        toggle = !toggle;
        delay(100);
    }

    // Отключаем звук
    noTone(BUZZER);

    // Выключаем таймер
    pomodoroState = false;
}

```

Интернет-нотификатор

Котика можно превратить в нотификатор для электронной почты. О том как это сделать, читайте в описании проекта [«Технокуб»](#)