Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Иркутский национальный исследовательский технический

университет»

Институт информационных технологий и анализа данных

**О Т Ч Ё Т**

|  |  |  |
| --- | --- | --- |
| о прохождении | | учебной практики |
|  | | (вид практики: учебная/производственная) |
| технологической (проектно-технологической) практики | | |
| (тип практики: технологическая/научно-исследовательская работа/преддипломная и др.) | | |
|  | | |
| в | ИРНИТУ | |
|  | (наименование профильной организации) | |

Обучающегося Амагаева С.С., ИСИБ-24-1

(ФИО, группа, подпись)

Руководитель практики от института ИТиАД

Кононенко Р.В., доцент института ИТиАД

(ФИО, должность, подпись)

Руководитель образовательной программы

Кононенко Р.В., доцент института ИТиАД

(ФИО, должность, подпись)

Оценка по практике

(ФИО, подпись, дата)

Содержание отчета на \_\_\_ стр.

Приложение к отчету на \_\_\_ стр.

Иркутск 2025

**Индивидуальное задание на прохождение**

**учебной практики: технологической (проектно-технологической) практики**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| для | Амагаева Станислава Сергеевича | | | | |
|  | (ФИО обучающегося полностью) | | | | |
| обучающегося | | 1 | курса | группы | ИСИБ-24-1 |

по направлению подготовки Информационные системы и технологии

профиль Интеллектуальные системы обработки информации и управления

Место прохождения практики: ИРНИТУ

Сроки прохождения практики с «16» июня 2025 г. по «29» июня 2025 г.

Цели и задачи прохождения практики:

Содержание практики, вопросы, подлежащие изучению:

Планируемые результаты практики:

Руководитель практики от

института ИТиАД

\_\_\_\_\_\_\_\_\_\_\_/ Кононенко Р.В. /

(подпись

**Согласовано:**

Руководитель ООП

\_\_\_\_\_\_\_\_\_\_\_/Кононенко Р.В./

(подпись

«\_\_\_»\_\_\_\_\_\_\_\_\_\_2025 г.

С настоящим индивидуальным заданием и с программой практики ознакомлен(а), задание принято к исполнению

\_\_\_\_\_\_\_\_\_\_«\_\_\_»\_\_\_\_\_\_\_\_2025 г.

(подпись

QR код на superjob QR код на hh.ru

<https://www.superjob.ru/resume/programmist-razrabotchik-55733586.html>

<https://hh.ru/resume/c26101d8ff0b294de00039ed1f644a464f654d>

**ДНЕВНИК**

прохождения практики

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| обучающегося | | | | | Амагаева Станислава Сергеевича,  ИСИБ-24-1 |
|  | | | | | (фамилия, имя, отчество, группа) |
|  | | | | |  |
| курс | | 1 | | | |
| направление | | | | Информатика и вычислительная техника | |
| профиль | | | Интеллектуальные системы обработки | | |
| информации и управления | | | | | |
| в | ИРНИТУ | | | | |
|  | (наименование профильной организации) | | | | |

Иркутск 2025

Руководителем практики от структурного подразделения назначен:

Кононенко Роман Владимирович, доцент института ИТиАД

(ФИО, должность)

**Рабочий график (план) прохождения практической подготовки**

(заполняется обучающимся)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № п/п | | Период  практики | Содержание выполненных работ | | Подпись руководителя практики от структурного подразделения |
| 1 | | 16.06.2025 | Решил задачу №1,  Решил задачу №2, | |  |
| 2 | | 17.06.2025 | Составил резюме на hh.ru и superjop.ru,  Решил задачу №3. | |  |
| 3 | | 18.06.2025 | Решил задачу №4,  Решил задачу №5. | |  |
| 4 | | 19.06.2025 | Решил задачу №6,  Решил задачу №7. | |  |
| 5 | | 20.06.2025 | Изучил теоретический материал для задачи №8,  Изучил теоретический материал для задачи №9. | |  |
| 6 | | 21.06.2025 – 23.06.2025 | Решил задачу №8,  Решил задачу №9.  Изучил теоретический материал для задачи №10.  Решил задачу №10. | |  |
| 7 | | 24.06.2025 | Группой ходили в филиал АО Со Еэс Иркутское РДУ. | |  |
| 8 | | 25.06.2025 | Изучил теоретический материал для задачи №11. | |  |
| 9 | | 26.06.2025 | Группой ходили в IT-компанию ISPsystem. | |  |
| 10 | | 27.06.2025 | Решил задачу №11. | |  |
| Дата фактического прибытия | | |  | |
| обучающегося в структурное подразделение | | | 16.06.2025 | |
| Дата фактического убытия | | |  | |
| обучающегося из структурного подразделения | | | 29.06.2025 | |

|  |  |
| --- | --- |
| Руководитель образовательной программы | Кононенко Р.В. |
|  | (ФИО, подпись) |
| Директор института | Говорков А.С. |
|  | (ФИО, подпись) |

**Содержание**

[Введение 6](#_Toc207647977)

[Задание №1 7](#_Toc207647978)

[Задание №2 10](#_Toc207647980)

[Задание №3 13](#_Toc207647980)

[Задание №4 16](#_Toc207647981)

[Задание №5 18](#_Toc207647982)

[Задание №6 20](#_Toc207647983)

[Задание №7 21](#_Toc207647984)

[Задание №8 22](#_Toc207647985)

[Задание №9 26](#_Toc207647986)

[Задание №10 30](#_Toc207647987)

[Отзыв о посещении филиала АО «СО ЕЭС» Иркутское РДУ 35](#_Toc207647989)

[Отзыв о посещении компании ISPsystem 36](#_Toc207647990)

[Задание №11 37](#_Toc207647987)

[Заключение 39](#_Toc207647991)

[Список литературы 40](#_Toc207647992)

**Введение**

Учебная практика была проведена на базе Иркутского национального исследовательского технического университета (ИРНИТУ) в период с 16 по 28 июня 2025 года. Главной целью являлось углубление и применение компетенций, сформированных в процессе освоения профильных дисциплин.

Основное внимание в ходе практики уделялось развитию практических навыков программирования, включая проектирование алгоритмов и работу с структурами данных. В рамках одиннадцати учебных заданий осваивались инструменты и методы, актуальные для современной IT-отрасли, что способствовало развитию способности самостоятельно анализировать и решать прикладные задачи.

Результатом прохождения практики стало не только закрепление теоретической базы, но и формирование профессионального подхода к реализации программных решений, что важно для дальнейшего обучения и будущей карьеры в сфере информационных технологий.

**Задание №1**

Незнайка в своей экспедиции на Луну оказался на вершине лунной горы. Спуск вниз опасен, поэтому он взял с собой карту склона горы, где числами обозначено, сколько минут требуется на этот участок маршрута. Спуск происходит сверху вниз на один из соседних участков. Пример наиболее короткого маршрута выделен красным цветом, сумма чисел = 10. Напишите программу, рассчитывающую минимальное время спуска (сумму чисел в пути с вершины до основания).

**Алгоритм программы:**

1. Генерация карты горы (func)

Создается двумерный массив vec, где каждая строка i имеет длину i (треугольная структура).

Каждый элемент заполняется случайным числом от 0 до 999.

Карта выводится на экран.

1. Основная логика (main)

Пользователь вводит m — высоту горы (количество уровней).

Генерируется карта (vec) с помощью func(m).

Начинается спуск с вершины (vec[0][0]), значение записывается в num и массив iii.

1. Динамический алгоритм спуска

На каждом уровне i выбирается минимальный из доступных соседей:

Если k = 0 (левый край), сравниваются vec[i][k] и vec[i][k + 1].

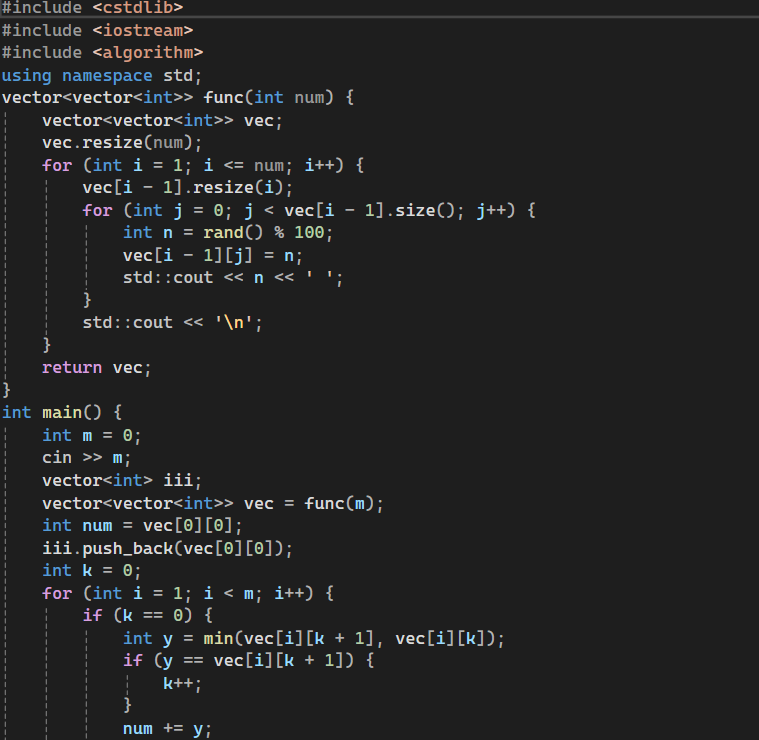
Если k в правом краю, сравниваются vec[i][k - 1] и vec[i][k].

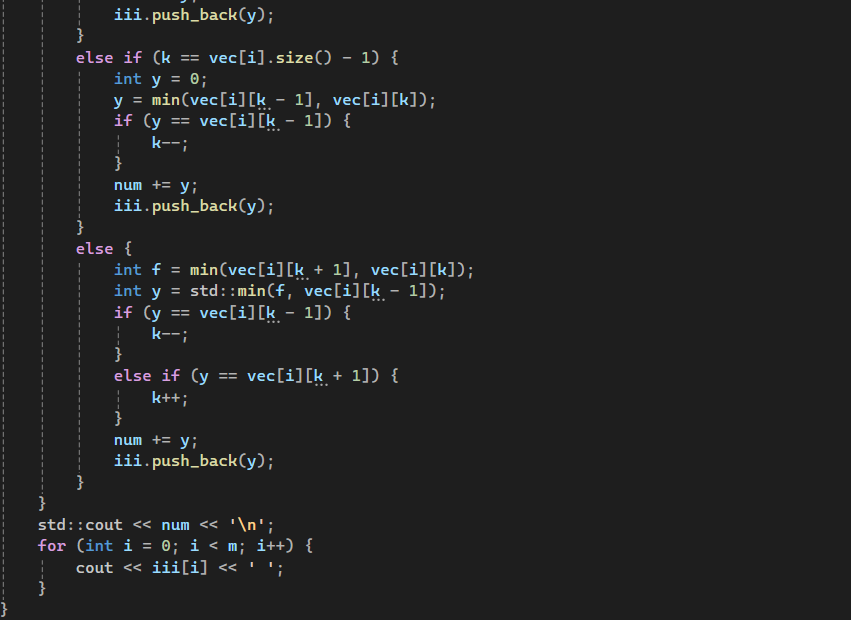
Иначе сравниваются все три соседа (k - 1, k, k + 1).

Выбранное минимальное значение добавляется в сумму num и массив iii.

Позиция k обновляется в зависимости от выбранного пути.

**Код программы:**

****

****

**Таблица тестов:**

|  |  |  |
| --- | --- | --- |
| Ввод | 4 | 5 |
| Вывод | 968  212 427  912 914 221  630 806 272 292  Минимальная сумма: 1888  Путь: 968 427 221 272 | 928  303 665  64 211 868  49 166 741 533  25 969 242 782 80  Минимальная сумма: 1369  Путь: 928 303 64 49 25 |

****

**Задача №2**

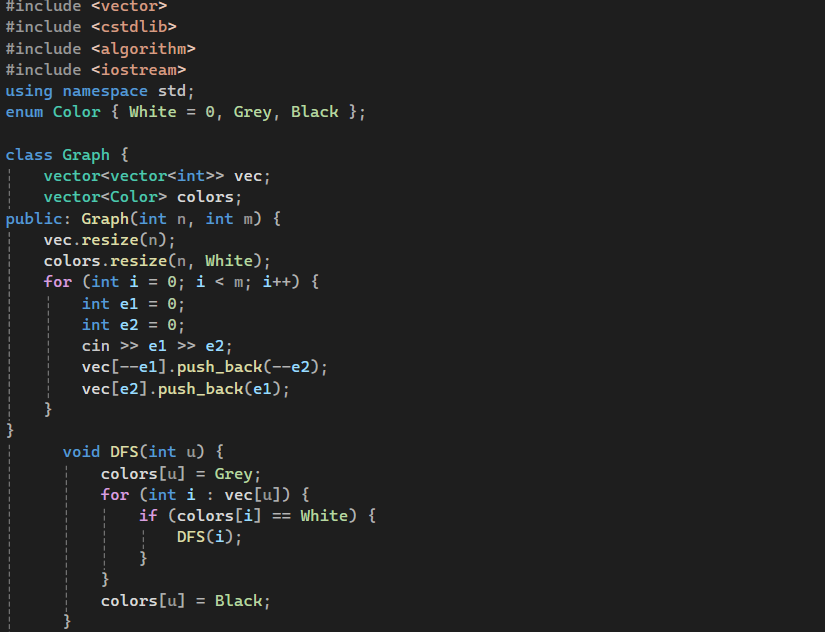
После метеоритной атаки компьютерная сеть для управления лунными заводами разбилась на части, нужно объединить её в единое целое. Каждый фрагмент сети представлен в виде ненаправленного графа.

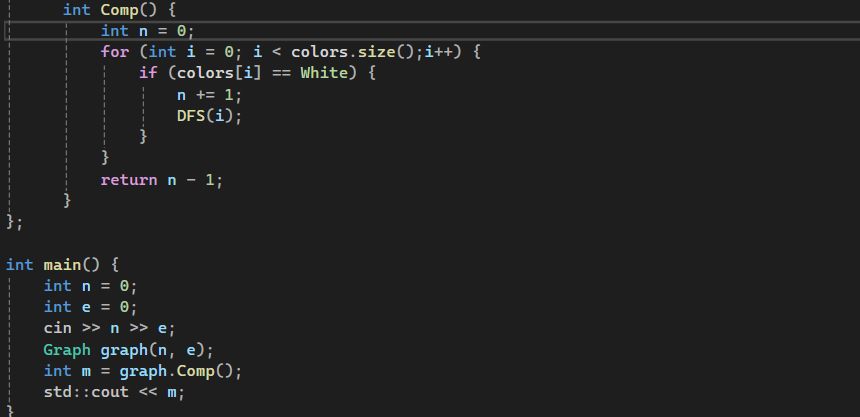
Вам известно общее число вершин графа (узлы сети, не более 1000) и набор рёбер (сохранившиеся линии связи, не более 1000).

Определите, какое минимальное число линий связи нужно дополнительно построить, чтобы сеть стала единой.

**Алгоритм программы:**

1. ***Представление сети в виде графа:*** вершины графа соответствуют узлам сети, а рёбра — существующим линиям связи между ними.
2. ***Поиск компонент связности:*** с помощью обхода графа в глубину (DFS) программа определяет количество компонент связности в графе. Компонента связности — это множество узлов, которые можно достичь друг из друга, следуя по существующим линиям связи.
3. *Подсчёт дополнительных линий связи:* если в графе существует более одной компоненты связности, то необходимо построить дополнительные линии связи для объединения этих компонент. Минимальное количество таких линий равно количеству компонент минус один.
4. *Вывод результата:* программа выводит минимальное количество линий связи, которые нужно добавить, чтобы сделать граф связным, то есть чтобы все узлы были соединены в единую сеть.

**Код программы:**

****

**Таблица тестов:**

|  |  |  |
| --- | --- | --- |
| Ввод | 10 6  1 2  2 8  4 10  5 9  6 10  7 9 | 7 4  1 2  2 3  4 5  6 7 |
| Вывод | 3 | 2 |



**Задача №3**

Иркутске раз в году наступает зима. Не смотря на то что событие это

довольно регулярное, оно всегда внезапно. Снег буквально

заваливает все улицы, не давая проехать на чём-то меньше трактора. В этом году терпение лопнуло и специальным указом был создан кризисный центр по борьбе с сугробами. Центру были переданы спутники, лазеры, метеорологические зонды и несколько десятков лопат.

Вам поручено возглавить отдел разведки снежной ситуации и быть способным чрезвычайно быстро отвечать на запросы центра. Сам город состоит из нескольких, расположенных подряд, улиц, каждая из которых абсолютна похожа на любую другую.

Информация о снеге передается вам в виде тройки чисел – 1 в качестве идентификатора события, уникального индекса улицы и количество миллиметров выпавшего снега.

* Запросы в свою очередь так же имеют вид тройки чисел – 2 в качестве идентификатора события, индекс улицы с которой нужно суммировать количество выпавшего снега и индекс улицы по которую нужно суммировать, крайние улицы должны быть включены.

**Алгоритм программы**

1. Инициализация

Создается массив snow размером n + 1 (улицы нумеруются с 1 до n), заполненный нулями.

Вектор results хранит ответы на запросы суммы.

1. Обработка k событий

Для каждого события считывается тройка чисел: command, a, b.

Если command == 1 (добавление снега

На улицу a добавляется b миллиметров снега: snow[a] += b.

Если command == 2 (запрос суммы снега)

1. Суммируется снег на улицах от a до b включительно.

Результат сохраняется в results.

1. Вывод ответов

После обработки всех запросов выводятся все сохраненные суммы.

**Код программы**



**Таблица тестов**

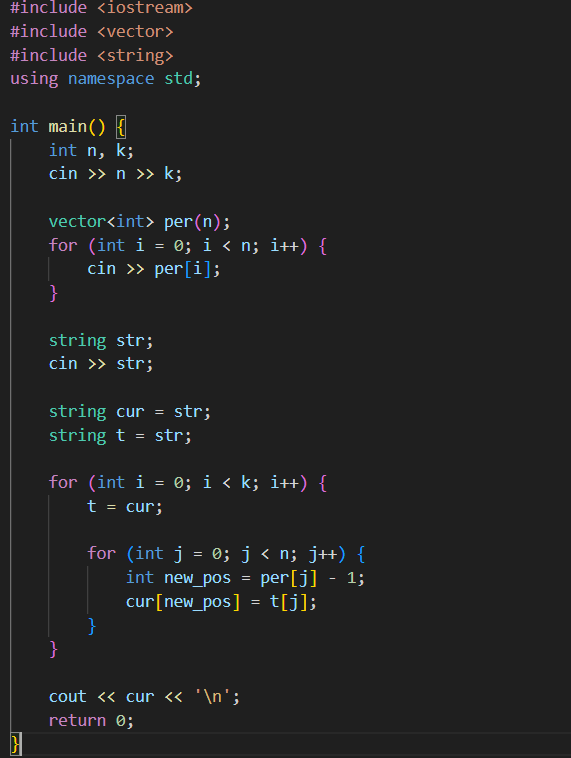
|  |  |  |
| --- | --- | --- |
| Ввод | 6 5  2 1 6  2 2 4  1 6 3  2 1 6 | 5 3  1 3 7  1 1 4  2 1 5 |
| Вывод | 0  2  5 | 11 |



**Задача №4**

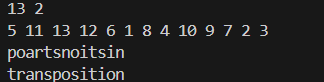
Перестановка P длины n − это упорядоченный набор, содержащий числа от 1 до n, каждое из которых входит в него ровно один раз. Например, перестановкой длины 13 является набор (5 11 13 12 6 1 8 4 10 9 7 2 3). Само название говорит о том, для чего предназначен этот объект. Например, можно при помощи перестановки букв зашифровать слово.

**Код программы**



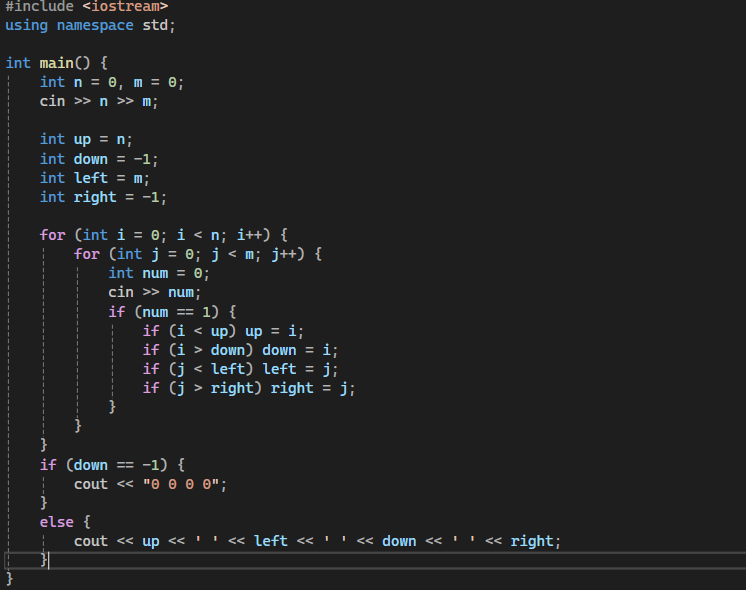
**Таблица тестов**

|  |  |  |
| --- | --- | --- |
| Ввод | 13 2  5 11 13 12 6 1 8 4 10 9 7 2 3  poartsnoitsin | 6 3  3 1 4 6 2 5  encode |
| Вывод | transposition | eodnce |



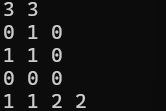
**Задача №5**

Дана матрица, состоящая из 1 и 0. Значениями 1 в матрице нарисована некоторая фигура. Необходимо определить координаты верхнего левого и нижнего правого углов параллельного осям ограничивающего прямоугольника, т.е. такого прямоугольника, минимального размера, в который фигура помещается полностью и при этом ни одна точка исходной фигуры не попадает на стороны прямоугольника.

**Код программы**

**Таблица тестов**

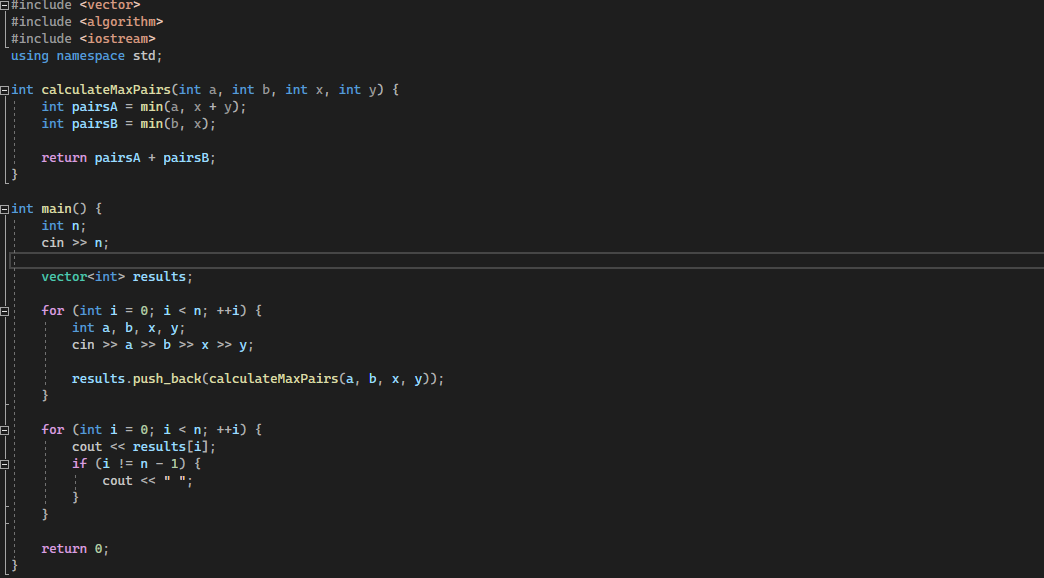
|  |  |  |
| --- | --- | --- |
| Ввод | 10 10  0 0 0 0 0 0 0 0 0 0  0 0 0 0 0 0 0 0 0 0  0 0 0 0 0 0 0 0 0 0  0 0 0 0 0 0 0 0 0 0  0 0 0 1 1 1 0 0 0 0  0 0 0 1 1 1 0 0 0 0  0 0 0 1 1 1 0 0 0 0  0 0 0 1 1 1 0 0 0 0  0 0 0 0 0 0 0 0 0 0  0 0 0 0 0 0 0 0 0 0 | 3 3  0 1 0  1 1 0  0 0 0 |
| Вывод | 3 2 8 6 | 1 1 2 2 |

****

**Задача №6**

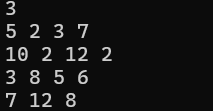
В школьном кружке робототехники есть два вида микроконтроллеров (условно тип A и тип B) и два вида модулей управления мотором (условно тип 1 и тип 2). Выяснилось, что контроллер типа B и модуль управления типа 2 несовместимы. Использование микроконтроллеров и модулей управления в других комбинациях возможно. Имеется a микроконтроллеров типа A, b микроконтроллеров типа B, x модулей управления типа 1 и y модулей типа 2. Определите, какое максимальное число работающих пар из микроконтроллера модуля управления мотором можно составить. Ваша программа должна ответить на n запросов.

**Код программы**



**Таблица тестов**

|  |  |
| --- | --- |
| Ввод | 3  5 2 3 7  10 2 12 2  3 8 5 6 |
| Вывод | 7 12 8 |

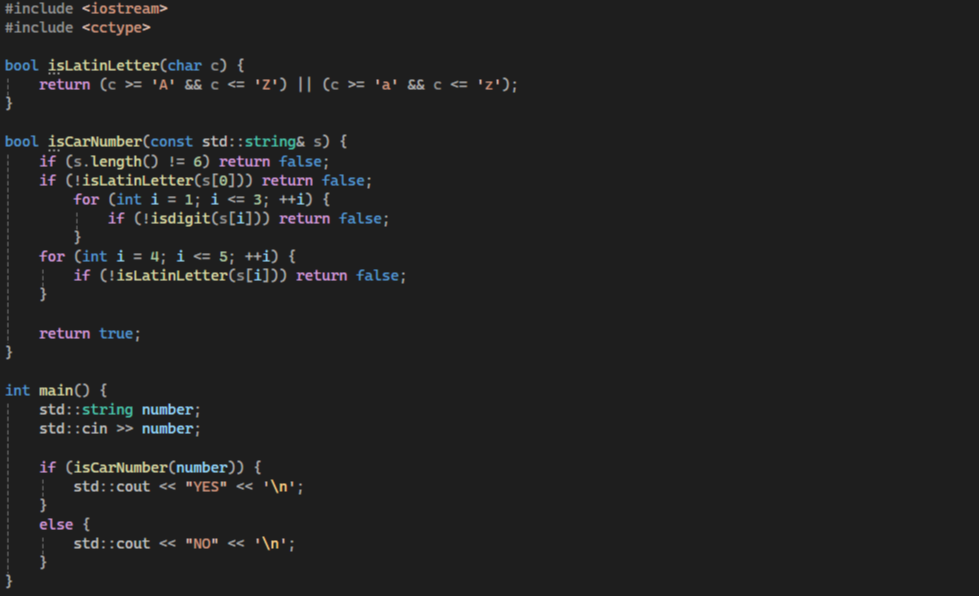


**Задача №7**

На компьютере работника автосервиса нашли файл с последовательностью автомобильных номеров, обслуживавшихся в этом автосервисе. Так как файл был поврежден, некоторые данные отображаются неверно. Нужно определить, какие из них остались невредимыми.

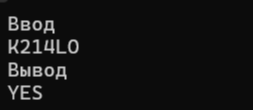
Автомобильным номером является строка из шести символов. Первый символ – заглавная латинская буква, далее следует 3 цифры, и после – две заглавные латинские буквы. Например, строка "P142EQ" является номером. Вам будет дана строка, состоящая из шести символов, необходимо ответить, является ли строка автомобильным номером.

**Код программы**



**Таблица тестов**

|  |  |  |
| --- | --- | --- |
| Ввод | K045OL | LL01KJ |
| Вывод | YES | NO |



**Задача №8**

Составить светодиодную матрицу размером не менее 8 на 8 светодиодов. На матрицу вывести инфографику с различными динамично меняющимися изображениями.

1. getPixelIndex(x, y)

Преобразует координаты (x,y) в номер светодиода на ленте (расположение "змейкой").

1. setPixelXY(x, y, color)

Зажигает светодиод в позиции (x,y) указанным цветом (формат strip.Color(R,G,B)).

1. showCross(duration, color)

Рисует зелёный крестик на 3 секунды (можно изменить цвет и время).

1. animatePulsingSquare(cycles)

Анимация пульсирующего квадрата:

Сначала растёт красный квадрат (2×2 → 8×8)

Затем сжимается зелёный квадрат (6×6 → 2×2)

Цикл повторяется 5 раз.

1. drawSquare(size, color)

Вспомогательная функция для рисования квадрата заданного размера и цвета.

1. Логика работы:

В setup() инициализируется LED-лента.

В loop() поочерёдно вызываются:

Отображение крестика (showCross())

Анимация квадрата (animatePulsingSquare())

**Код программы:**

#include <Adafruit\_NeoPixel.h>

#define LED\_PIN     6       // Пин для ленты NeoPixel

#define LED\_COUNT   64      // 8x8 = 64 светодиода

#define BRIGHTNESS  100     // Яркость (0-255)

#define WIDTH       8       // Ширина матрицы

#define HEIGHT      8       // Высота матрицы

Adafruit\_NeoPixel strip(LED\_COUNT, LED\_PIN, NEO\_GRB + NEO\_KHZ800);

// Функция для преобразования координат (x,y) в индекс светодиода (змейкой)

int getPixelIndex(int x, int y) {

    return (y % 2 == 0) ? y \* WIDTH + x : y \* WIDTH + (WIDTH - 1 - x);

}

void setPixelXY(int x, int y, uint32\_t color) {

    if (x >= 0 && x < WIDTH && y >= 0 && y < HEIGHT) {

        strip.setPixelColor(getPixelIndex(x, y), color);

    }

}

void setup() {

    strip.begin();

    strip.show();

    strip.setBrightness(BRIGHTNESS);

}

void loop() {

    showCross(3, strip.Color(0, 255, 0));  // Зелёный крестик на 3 секунды

    animatePulsingSquare(5);               // Анимация пульсирующего квадрата (5 циклов)

}

// Функция для отрисовки крестика (без изменений)

void showCross(unsigned long duration, uint32\_t color) {

    uint8\_t cross[8][8] = {

      {1,0,0,0,0,0,0,1},

      {0,1,0,0,0,0,1,0},

      {0,0,1,0,0,1,0,0},

      {0,0,0,1,1,0,0,0},

      {0,0,0,1,1,0,0,0},

      {0,0,1,0,0,1,0,0},

      {0,1,0,0,0,0,1,0},

      {1,0,0,0,0,0,0,1}

    };

    unsigned long start = millis();

    while (millis() - start < duration \* 1000) {

        for (int y = 0; y < 8; y++) {

            for (int x = 0; x < 8; x++) {

                setPixelXY(x, y, cross[y][x] ? color : 0);

            }

        }

        strip.show();

        delay(50);

    }

}

// Анимация пульсирующего квадрата (простое изменение размера)

void animatePulsingSquare(int cycles) {

    for (int cycle = 0; cycle < cycles; cycle++) {

        // Плавное увеличение квадрата (от 2x2 до 8x8)

        for (int size = 2; size <= 8; size += 2) {

            drawSquare(size, strip.Color(255, 0, 0)); // Красный квадрат

            delay(300);

            strip.clear();

        }

        // Плавное уменьшение квадрата (с зелёным цветом)

        for (int size = 6; size >= 2; size -= 2) {

            drawSquare(size, strip.Color(0, 255, 0)); // Зелёный квадрат

            delay(300);

            strip.clear();

        }

    }

}

// Рисуем квадрат с центром в середине матрицы

void drawSquare(int size, uint32\_t color) {

    int startX = (WIDTH - size) / 2;

    int startY = (HEIGHT - size) / 2;

    for (int x = startX; x < startX + size; x++) {

        for (int y = startY; y < startY + size; y++) {

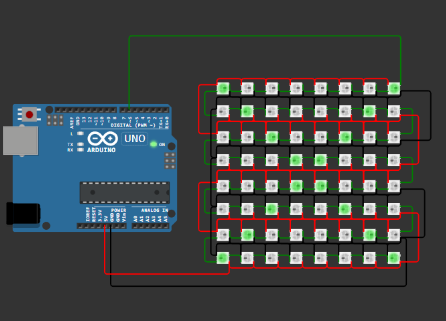
            setPixelXY(x, y, color);

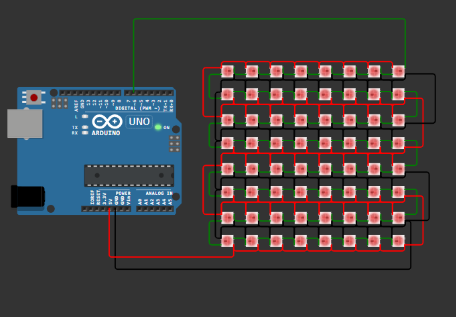
        }

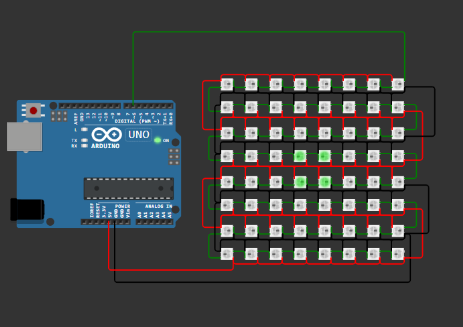
    }

    strip.show();

}

****

****

****

**Задача №9**

1. Собрать схему имитирующую работу автоматических дверей
2. Подобрать номинал резисторов для светодиодов
3. Написать программу для управления процессом работы автоматических дверей.

Зеленый светодиод – двери отрываются.

Красный светодиод ¬– двери закрываются.

Фоторезистор имитируют процесс приближения-удаления человека от дверей.

Программа управляет светодиодами, имитируя работу автоматических дверей:

• При низком уровне освещённости (когда фоторезистор обнаруживает человека рядом), загорается зелёный светодиод (сигнализируя об открытии дверей).

• При высоком уровне освещённости (когда человека нет), активируется красный светодиод (указывая на закрытие дверей).

**Код программы:**

#define GREEN\_LED\_PIN 9       // Зелёный светодиод - двери открыты

#define RED\_LED\_PIN 5         // Красный светодиод - двери закрыты

#define SENSOR\_PIN A0         // Фоторезистор

// Время, на которое двери остаются открытыми (3 секунды)

const unsigned long DOOR\_OPEN\_DURATION = 3000;

// Пороговое значение для определения приближения (512 - среднее значение)

const int PRESENCE\_THRESHOLD = 512;

// Состояние дверей

enum DoorState {

  DOOR\_CLOSED,

  DOOR\_OPEN,

  DOOR\_HOLD\_OPEN

};

DoorState currentState = DOOR\_CLOSED;

unsigned long doorOpenTime = 0; // Время, когда двери были открыты

void setup() {

  pinMode(RED\_LED\_PIN, OUTPUT);

  pinMode(GREEN\_LED\_PIN, OUTPUT);

  Serial.begin(9600);

  // Инициализация - двери закрыты

  setDoorsClosed();

}

void loop() {

  int sensorValue = analogRead(SENSOR\_PIN);

  Serial.print("Значение с фоторезистора: ");

  Serial.println(sensorValue);

  switch(currentState) {

    case DOOR\_CLOSED:

      handleClosedState(sensorValue);

      break;

    case DOOR\_OPEN:

      handleOpenState(sensorValue);

      break;

    case DOOR\_HOLD\_OPEN:

      handleHoldOpenState(sensorValue);

      break;

  }

  delay(200); // Небольшая задержка для стабильности

}

void handleClosedState(int sensorValue) {

  if (sensorValue < PRESENCE\_THRESHOLD) {

    // Обнаружено присутствие - открываем двери

    setDoorsOpen();

    doorOpenTime = millis();

    currentState = DOOR\_OPEN;

    Serial.println("Обнаружен человек - двери открываются");

  }

}

void handleOpenState(int sensorValue) {

  // Проверяем, истекло ли время открытия

  if (millis() - doorOpenTime >= DOOR\_OPEN\_DURATION) {

    if (sensorValue < PRESENCE\_THRESHOLD) {

      // Человек всё ещё рядом - остаёмся открытыми

      currentState = DOOR\_HOLD\_OPEN;

      Serial.println("Человек всё ещё рядом - двери остаются открытыми");

    } else {

      // Человек ушёл - закрываем двери

      setDoorsClosed();

      currentState = DOOR\_CLOSED;

      Serial.println("Человек ушёл - двери закрываются");

    }

  }

}

void handleHoldOpenState(int sensorValue) {

  if (sensorValue >= PRESENCE\_THRESHOLD) {

    // Человек ушёл - закрываем двери

    setDoorsClosed();

    currentState = DOOR\_CLOSED;

    Serial.println("Человек ушёл - двери закрываются");

  }

}

void setDoorsOpen() {

  digitalWrite(RED\_LED\_PIN, LOW);

  digitalWrite(GREEN\_LED\_PIN, HIGH);

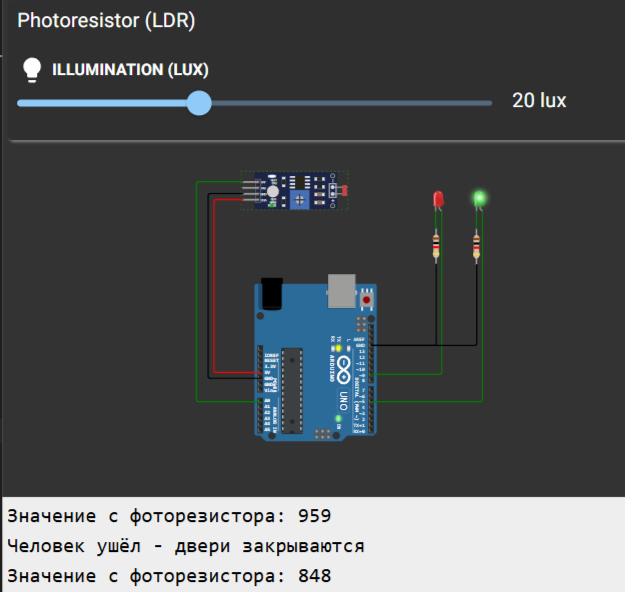
}

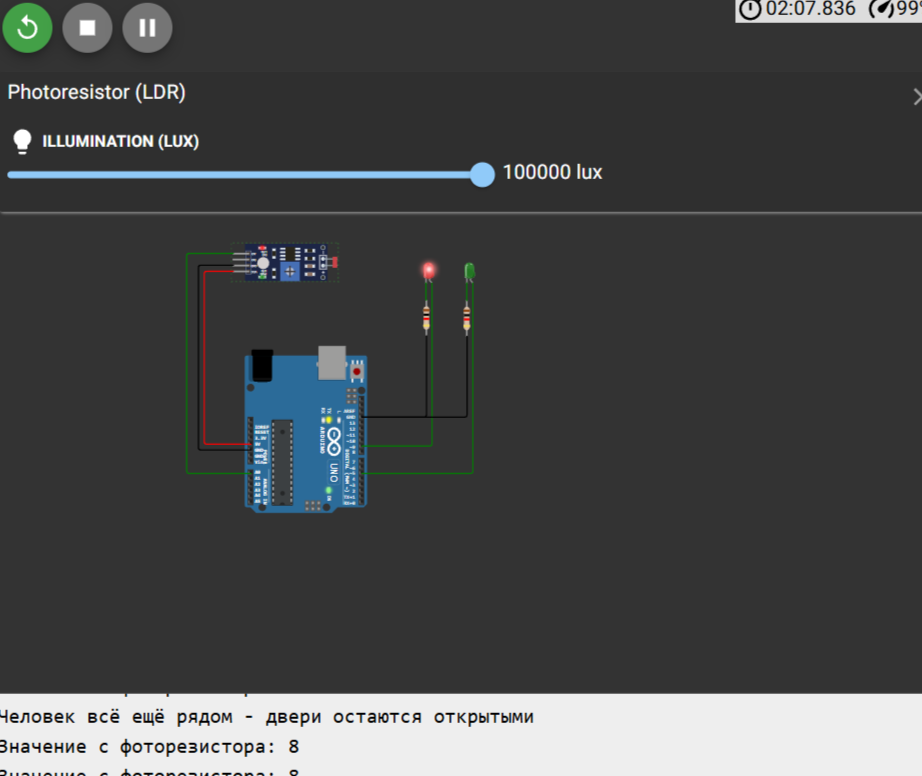
void setDoorsClosed() {

  digitalWrite(RED\_LED\_PIN, HIGH);

  digitalWrite(GREEN\_LED\_PIN, LOW);

}



****

**Задача №10**

1. Собрать схему подключения сервопривода
2. Написать программу для управления сервоприводом через последовательный порт

Логика работы программы: запрашивается угол поворота сервопривода, если он отличен от того, на который повернут привод, то плавно повернуть до указанного. Программа работает в цикле, с возможностью постоянно изменять угол поворота.

1. Инициализация системы:

Производится настройка сервопривода с установкой начального положения в 90 градусов (нейтральная позиция)

Инициализируется последовательный порт для мониторинга и управления с возможностью вывода служебных сообщений

2. Мониторинг управляющих команд:

В основном цикле программа постоянно проверяет входной буфер последовательного порта

При обнаружении корректного значения угла (в диапазоне 0-180 градусов), отличного от текущей позиции, инициируется процедура поворота

3. Алгоритм плавного перемещения:

Программа вычисляет разницу между текущим и целевым углом

Осуществляется постепенное изменение положения с небольшим шагом

На каждом шаге производится кратковременная задержка для создания эффекта плавности

4. Циклическая обработка:

После завершения перемещения система возвращается в режим ожидания новых команд

Процесс повторяется бесконечно, обеспечивая возможность оперативного изменения положения сервопривода

**Код программы**

#include <Servo.h>

// Константы

const byte SERVO\_PIN = 3;       // Пин сервопривода

const int DEFAULT\_ANGLE = 90;   // Стартовое положение

const int MIN\_ANGLE = 0;        // Минимальный угол

const int MAX\_ANGLE = 180;      // Максимальный угол

const int DELAY\_STEP = 15;      // Задержка между шагами (мс)

// Глобальные переменные

Servo myServo;

int currentAngle = DEFAULT\_ANGLE;

int targetAngle = DEFAULT\_ANGLE;

void setup() {

  Serial.begin(9600);

  while (!Serial); // Ожидание подключения Serial (для Arduino Leonardo)

  myServo.attach(SERVO\_PIN);

  myServo.write(DEFAULT\_ANGLE);

  printWelcomeMessage();

}

void loop() {

  handleSerialInput();

  updateServoPosition();

}

void handleSerialInput() {

  if (Serial.available()) {

    String input = Serial.readStringUntil('\n');

    input.trim();

    if (input.length() > 0) {

      processAngleInput(input);

    }

  }

}

void processAngleInput(String input) {

  if (isValidNumber(input)) {

    int angle = input.toInt();

    if (angle >= MIN\_ANGLE && angle <= MAX\_ANGLE) {

      if (angle != currentAngle) {

        targetAngle = angle;

        Serial.print("Принят новый угол: ");

        Serial.println(targetAngle);

      } else {

        Serial.println("Сервопривод уже в этом положении");

      }

    } else {

      Serial.println("Ошибка: угол должен быть от 0 до 180");

    }

  } else {

    Serial.println("Ошибка: введите целое число");

  }

}

bool isValidNumber(String str) {

  for (byte i = 0; i < str.length(); i++) {

    if (!isDigit(str.charAt(i))) {

      return false;

    }

  }

  return str.length() > 0;

}

void updateServoPosition() {

  if (currentAngle != targetAngle) {

    int step = (targetAngle > currentAngle) ? 1 : -1;

    currentAngle += step;

    myServo.write(currentAngle);

    delay(DELAY\_STEP);

  }

}

void printWelcomeMessage() {

  Serial.println("Система управления сервоприводом");

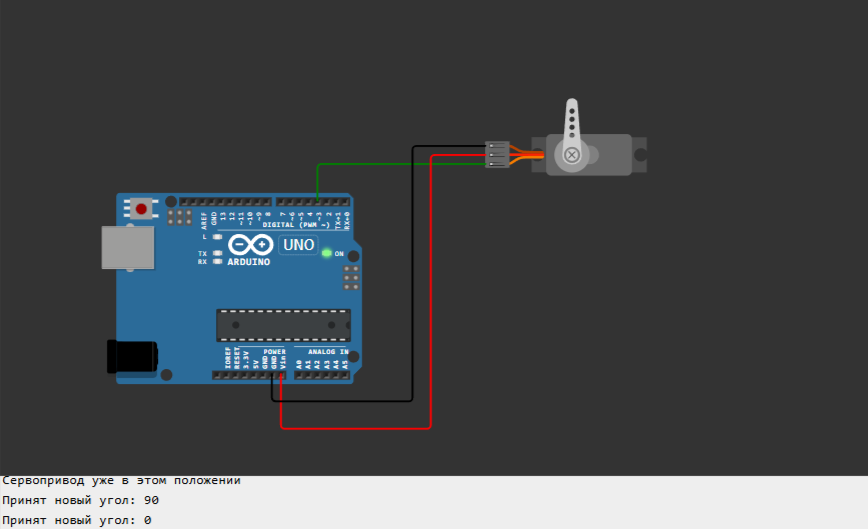
  Serial.println("--------------------------------");

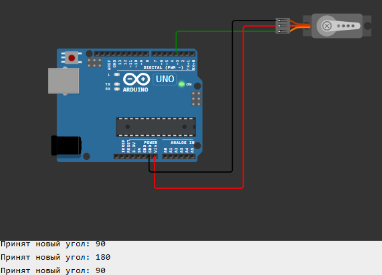
  Serial.println("Введите угол от 0 до 180 градусов");

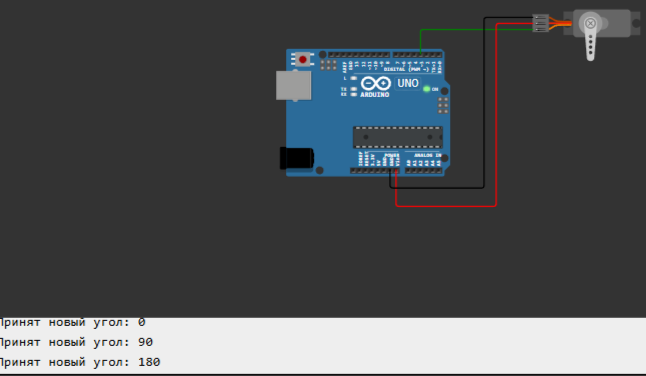
  Serial.println("Текущее положение: " + String(DEFAULT\_ANGLE));

  Serial.println("--------------------------------");

}



****



# 

# **Отзыв о посещении филиала АО «СО ЕЭС» Иркутское РДУ**

В ходе учебной практики наша группа побывала в Иркутском региональном диспетчерском управлении, которое является частью АО «Системный оператор ЕЭС». Это предприятие выступает ключевым звеном в поддержании бесперебойной работы энергокомплекса Иркутской области.

В фокусе деятельности РДУ — оперативное управление региональной энергосистемой: координация функционирования электростанций и сетевых объектов, а также текущее регулирование выработки и распределения электроэнергии.

Особое внимание во время визита было уделено растущей роли диджитал решений в энергетике. Для решения этих задач в компании создан IT-отдел, который занимается созданием и внедрением интеллектуальных систем для поддержки диспетчеров. Среди них — автоматизированный информационный комплекс для аналитики и предоставления оперативных данных, а также передовая симуляционная платформа, предназначенная для моделирования поведения энергообъектов в различных сценариях.

Этот визит дал нам ясное представление о принципах диспетчеризации, критической важности IT-инфраструктуры в современной энергетике и высоком уровне ответственности, который лежит на компании за обеспечение энергетической стабильности региона.

**Отзыв о посещении компании ISPsystem**

На практике мы с группой съездили в компанию ISPsystem. Они разрабатывают ПО для управления IT-инфраструктурой. Главное, что мы поняли — они сами придумывают и делают свои программы, а не работают на заказ или на чужом хостинге.

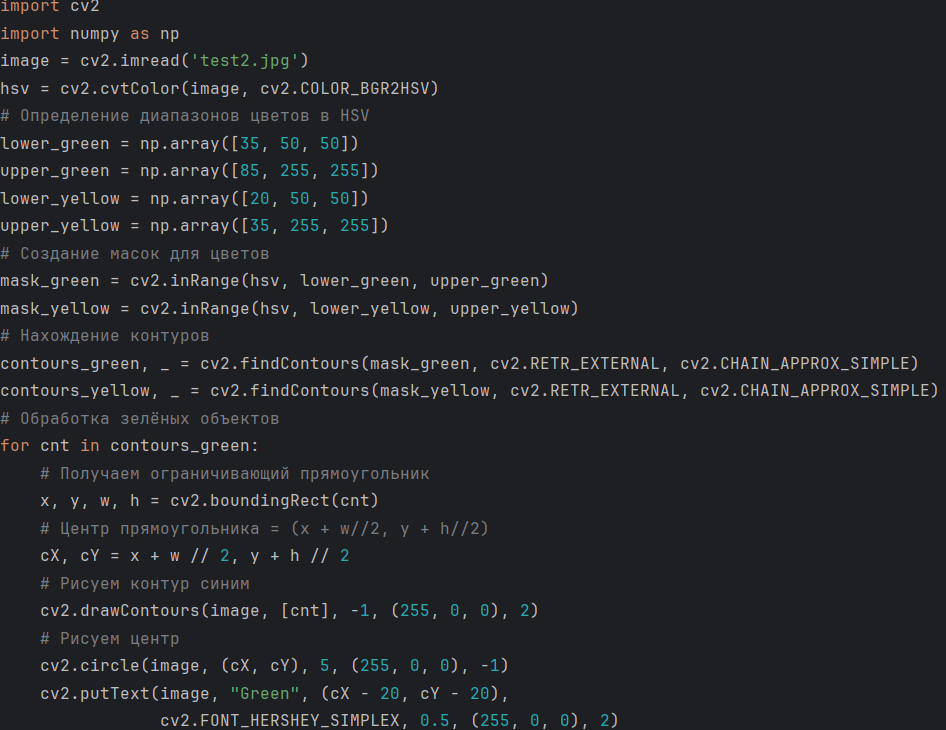
Мне очень понравились люди, которые там работают. Все такие классные, улыбаются, видно, что им самому нравится их работа и они гордятся тем, что делают. С ними было легко и интересно разговаривать.

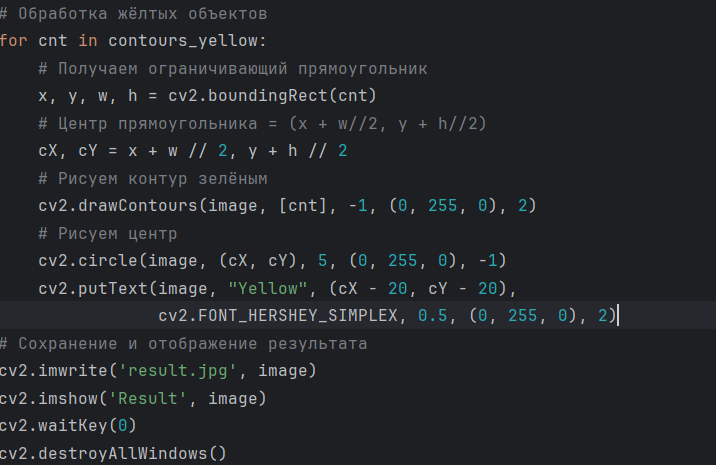
Ещё нам рассказали, кого они ищут к себе в команду. Оказалось, что нужны не только программисты (бэкенд, фронтенд, тестировщики), но и много кто ещё: дизайнеры, менеджеры проектов, маркетологи, продажники, те, кто пишет инструкции, и даже те, кто переводит программы на другие языки. То есть попасть туда можно с самыми разными навыками.

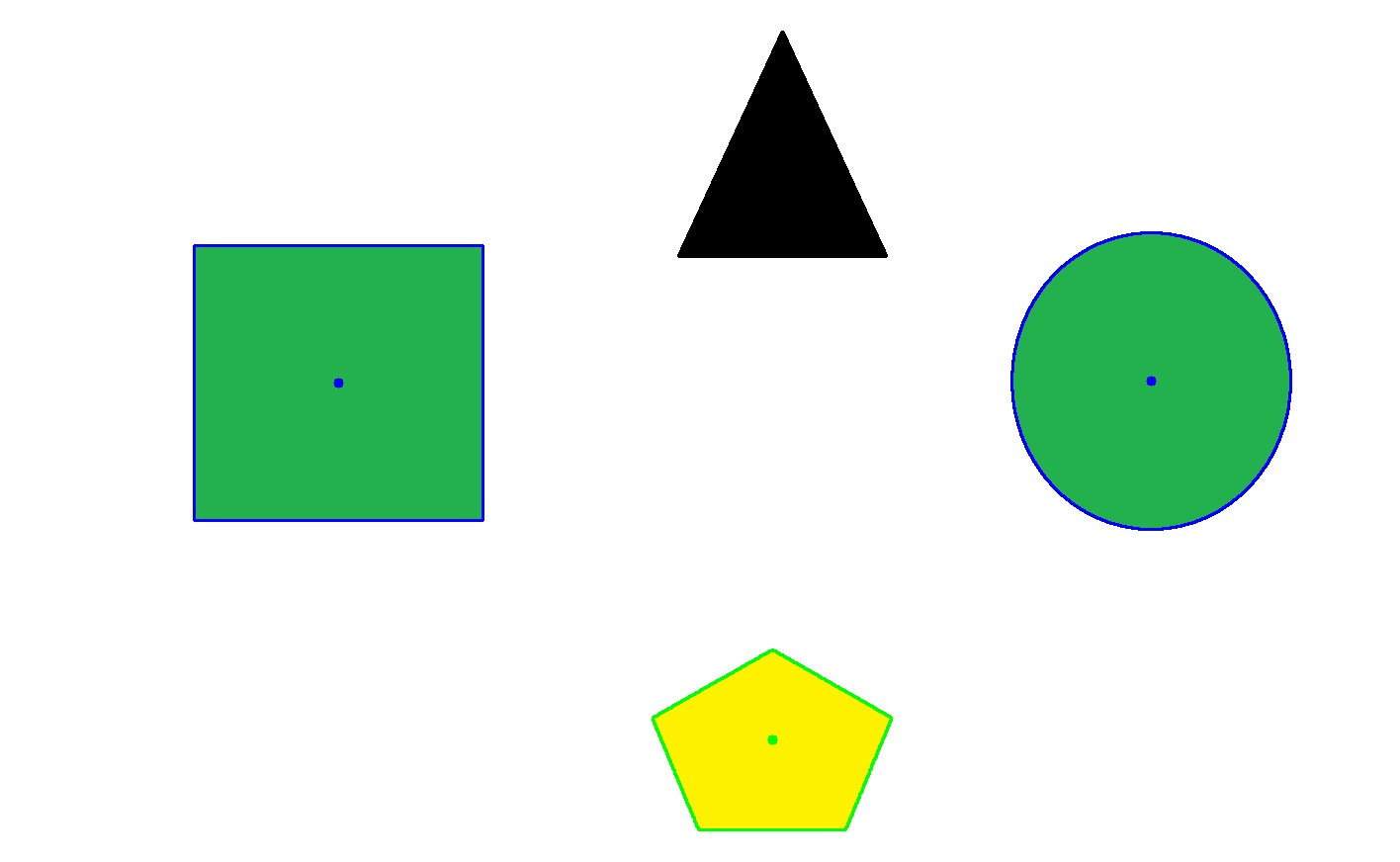
В общем, мне там очень понравилось. Было видно, что в компании дружная атмосфера и все реально горят своим делом. Теперь я лучше представил, как выглядит работа в нормальной IT-компании изнутри, и куда можно стремиться.

**Задача №11**

Найдите все зеленые и желтые объекты на изображении. Найдите центры всех зеленых и желтых объектов. Выделите границы зеленых объектов синей рамкой. Выделите границы желтых объектов зеленой рамкой.







**Заключение**

В период практики мне удалось применить и значительно углубить теоретические знания, полученные за время изучения курсов “C++” и “Алгоритмы и структуры данных”. Основной практической задачей стало развитие навыков программирования на языках программирования (я пользовался C++ и Python) и освоение работы с аппаратной частью на платформе Arduino. Работа с различными модулями и датчиками позволила наглядно изучить взаимодействие между программным кодом и физическими устройствами. Отдельным и крайне интересным направлением работы стало знакомство с основами машинного зрения. В рамках этого блока я разрабатывал программы на Python, используя библиотеки OpenCV и NumPy. Это дало возможность освоить на практике базовые алгоритмы обработки изображений и методы распознавания объектов по цветовым признакам.

Важным дополнением к технической части стали экскурсии в компании ISPsystem и Иркутское РДУ. Они предоставили ценный взгляд изнутри на работу современных IT- и энергетических предприятий, их технологический стек и подходы к управлению проектами.

Весь приобретенный опыт не только систематизировал пройденный материал, но и сформировал четкое понимание, как полученные в университете знания применяются в реальных производственных задачах. Уверен, что это будет фундаментом для дальнейшего обучения и поможет в будущей профессиональной деятельности.

**Список литературы**

1. Wokwi Arduino Simulator: онлайн-симулятор Arduino-проектов

URL: <https://wokwi.com/arduino>

1. OpenCV: компьютерное зрение на Python

URL: <https://habr.com/ru/companies/otus/articles/849136/>

1. Гайд по адресной светодиодной ленте

URL: <https://alexgyver.ru/ws2812_guide/>

1. Гайд по адресным светодиодным матрицам

URL: https://alexgyver.ru/matrix\_guide/