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 Просекина Филиппа Романовича, ИСИб-24-1

(ФИО, группа, подпись)

Руководитель практики от института ИТиАД

Кононенко Роман Владимирович, доцент института ИТиАД

Ссылка на резюме на сайте https://www.superjob.ru/

(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.

Ссылка на резюме на сайте https://www.hh.ru/

Иркутск 2025

**Задание №1**

**Постановка задачи:**

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. Напишите программу, рассчитывающую минимальное время спуска (сумму чисел в пути с вершины до основания).

Формат входных данных :

В первой строке дано целое число N - высота пирамиды, далее следуют N строк из чисел, разделённых пробелом (в каждой строке на 1 число больше, чем в предыдущей)

Формат выходных данных:

Сумма чисел в пути с вершины до основания (одно число)

Последовательность участков маршрута (числа, разделённые пробелом

**Ход решения:**

#include <iostream>

#include <cstdlib>

#include <ctime>

#include <vector>

int main() {

    std::cout << "Введите число ступеней горы:" << '\n';

    int count = 0;

    std::cin >> count;

    std::srand(std::time(0));

    std::vector<std::vector<int>> step(count);

    int way\_sum = 0;

    std::vector<int> way(count);

    step[0].resize(1);

    step[0][0] = std::rand() % 1000 + 1;

    int idx\_last\_min = 0;

    int min = 0;

    way[0] = step[0][0];

    for (int i = 1; i < count; i++) {

        step[i].resize(i + 1);

        for (int j = 0; j < i + 1; j++) {

            step[i][j] = std::rand() % 1000 + 1;

        }

        min = step[i][idx\_last\_min];

        if (step[i][idx\_last\_min + 1] < min) {

            min = step[i][idx\_last\_min + 1];

            idx\_last\_min += 1;

        }

        way\_sum += min;

        way[i] = min;

    }

    std::cout << "Время кратчайшего пути: " << '\n';

    std::cout << way\_sum << '\n';

    std::cout << "Кратчайший путь: " << '\n';

    for (int i = 0; i < count; i++) {

        std::cout << way[i] << ' ';

    }

    std::cout << '\n';

    std::cout << "Карта горы:\n" ;

    for (int i = 0; i < count; i++) {

        for (int j = 0; j < i + 1; j++) {

            std::cout << step[i][j] << ' ';

        }

        std::cout << '\n';

    }

}

**Результат:**

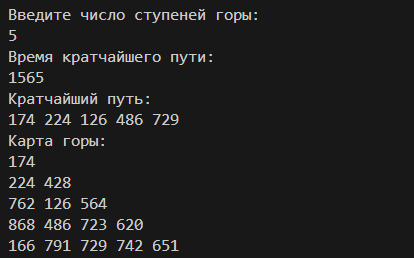


Рисунок 1

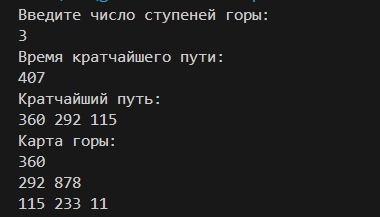


Рисунок 2

**Задание №2**

**Постановка задачи:**

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.

Формат входных данных:

В первой строке дано целое число N - количество узлов сети и M - число линий связи. Далее следуют M строк из чисел, разделённых пробелом (узлы, которые связывает данная линия)

Формат выходных данных:

Число необходимых линий связи (одно число)

**Ход решения:**

#include <iostream>

#include <vector>

#include <cstdlib>

#include <ctime>

#include <algorithm>

using namespace std;

void dfs(int v, vector<bool>& visited, vector<vector<int>>& graph) {

    visited[v] = true;

    for (int u : graph[v]) {

        if (!visited[u]) {

            dfs(u, visited, graph);

        }

    }

}

int main() {

    srand(time(0));

    int N, M = 0;

    cout << "Введите число вершин: " << '\n';

    cin >> N;

    cout << "Введите число рёбер: " << '\n';

    cin >> M;

    vector<vector<int>> graph(N);

    vector<bool> edge\_exists(N \* N, false);

    int edges\_created = 0;

    while (edges\_created < M) {

        int u = rand() % N;

        int v = rand() % N;

        if (u != v && !edge\_exists[u \* N + v]) {

            graph[u].push\_back(v);

            graph[v].push\_back(u);

            edge\_exists[u \* N + v] = true;

            edge\_exists[v \* N + u] = true;

            edges\_created++;

        }

    }

    cout << "Рёбра в графе:" << endl;

    for (int u = 0; u < N; ++u) {

        for (int v : graph[u]) {

            if (u < v) {

                cout << u << " " << v << endl;

            }

        }

    }

    vector<bool> visited(N, false);

    int components = 0;

    for (int i = 0; i < N; ++i) {

        if (!visited[i]) {

            dfs(i, visited, graph);

            components++;

        }

    }

    cout << "Минимальное число рёбер для объединения сети: ";

    cout << components - 1 << endl;

    return 0;

}

**Результат:**

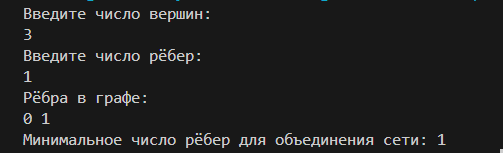


Рисунок 3

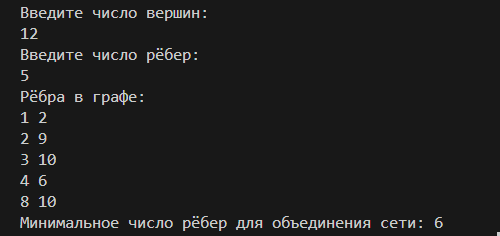
****

Рисунок 4

**Задача №3**

**Постановка задачи:**

В 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.

Формат входных данных:

Первая строка входных данных содержит два целых числа – n (1 или больше) и k (0 или больше) это количество чисел в массиве и количество запросов соответственно.

Следующие k строк содержат:

* либо 1 i x – Учетная информация о количестве, выпавшего на улице i (больше 0) x миллиметров снега.
* либо 2 u r – Запрос на подсчет количества снега на улицах от u до r (u и r больше 0 и могут быть равны друг другу)

Формат выходных данных:

* На каждый запрос второго типа надо вывести единственное число – суммарное выпавшего на них снега с момента начала наблюдения

**Ход решения:**

#include <iostream>

#include <vector>

#include <cstdlib>

#include <ctime>

using namespace std;

int main() {

    srand(time(0));

    int n, k;

    cin >> n >> k;

    vector<int> snow(n + 1, 0);

    vector<vector<int>> queries;

    for (int i = 0; i < k; i++) {

        int type = rand() % 2 + 1;

        int a = rand() % n + 1;

        int b;

        if (type == 1) {

            b = rand() % 10 + 1;

            queries.push\_back({type, a, b});

        } else {

            b = rand() % n + 1;

            if (a > b) swap(a, b);

            queries.push\_back({type, a, b});

        }

    }

    cout << "(Автоматически сгенерированные) Запросы:" << endl;

    for (const auto& q : queries) {

        cout << q[0] << " " << q[1] << " " << q[2] << endl;

    }

    cout << endl;

    cout << "Результаты:" << endl;

    for (const auto& q : queries) {

        int type = q[0];

        int a = q[1];

        int b = q[2];

        if (type == 1) {

            snow[a] += b;

        } else {

            int sum = 0;

            for (int i = a; i <= b; i++) {

                sum += snow[i];

            }

            cout << sum << endl;

        }

    }

    return 0;

}

**Результат:**

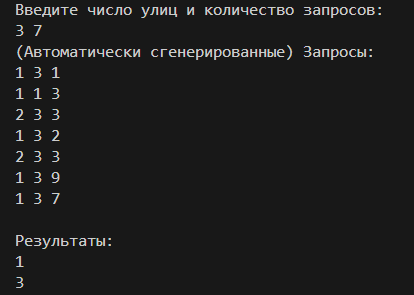


Рисунок 5

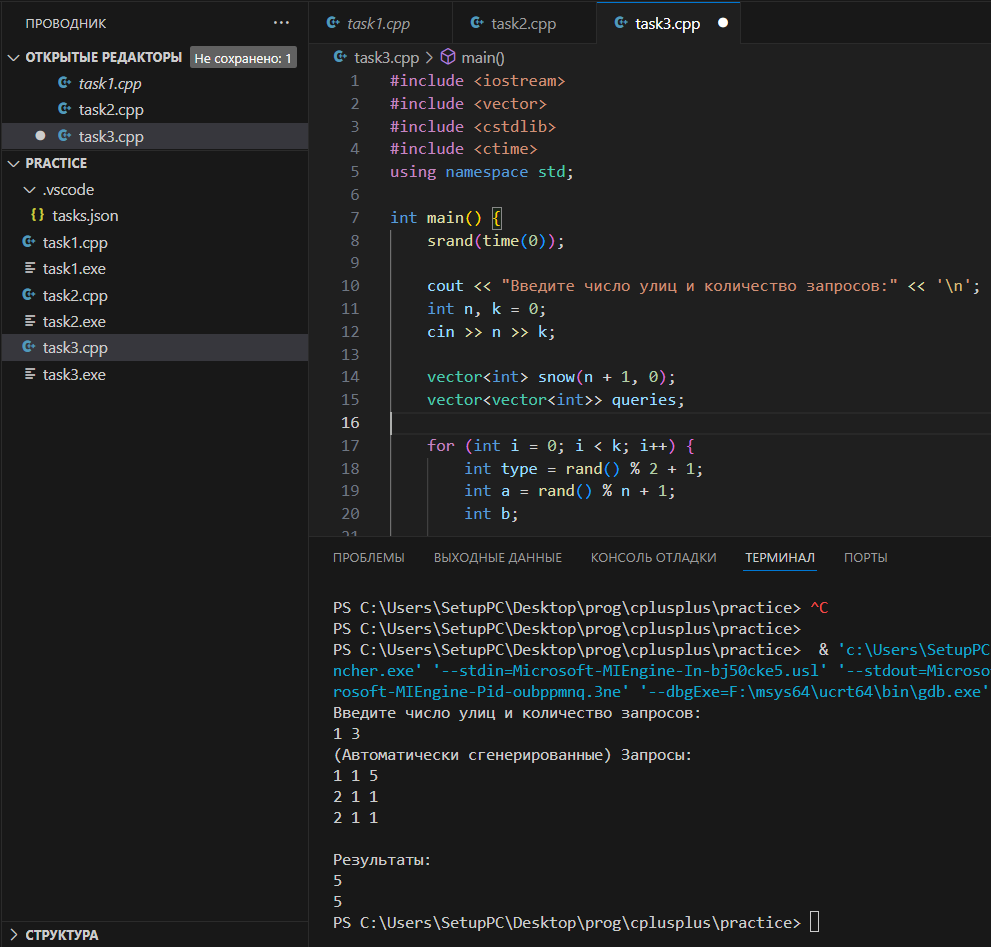


Рисунок 6

**Задача №4**

**Постановка задачи:**

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. Для примера возьмем приведенную выше перестановку и слово transposition, которое состоит тоже из 13 букв. Далее, следуя перестановке, на первую позицию поставим пятую букву слова, на вторую − одиннадцатую букву и так далее. В итоге получим sinoptsntiora. К этому слову снова применим эту же перестановку и получим poartsnoitsin. Повторив эти стадии шифрования k раз, получим зашифрованное сообщение.

Вам дано зашифрованное таким образом слово, шифрующая перестановка P и число k. Необходимо восстановить слово.

Формат входных данных:

Первая строка входных данных содержит 2 числа – n и k (1 или больше, могут быть равны). Следующая строка содержит перестановку длиной n, числа разделяются пробелом. Третья строка содержит зашифрованное слово длиной n.

Формат выходных данных:

Вывести одну строку − исходное слово

**Ход решения:**

#include <iostream>

#include <vector>

#include <string>

std::vector<int> inversePermutation(const std::vector<int>& p) {

    std::vector<int> inv(p.size());

    for (size\_t i = 0; i < p.size(); ++i) {

        inv[p[i] - 1] = i + 1;

    }

    return inv;

}

std::vector<int> permutationPower(const std::vector<int>& p, int power) {

    std::vector<int> result(p.size());

    for (size\_t i = 0; i < p.size(); ++i) {

        int current = i + 1;

        for (int j = 0; j < power; ++j) {

            current = p[current - 1];

        }

        result[i] = current;

    }

    return result;

}

std::string applyPermutation(const std::string& s, const std::vector<int>& p) {

    std::string result(s.size(), ' ');

    for (size\_t i = 0; i < s.size(); ++i) {

        result[i] = s[p[i] - 1];

    }

    return result;

}

int main() {

    int n, k;

    std::cout << "Введите длину слова и число шифрований: ";

    std::cin >> n >> k;

    std::vector<int> p(n);

    std::cout << "Введите перестановку из " << n << " чисел:\n";

    for (int i = 0; i < n; ++i) {

        std::cin >> p[i];

    }

    std::string encrypted;

    std::cout << "Введите зашифрованное слово: ";

    std::cin >> encrypted;

    std::vector<int> invP = inversePermutation(p);

    std::vector<int> invPk = permutationPower(invP, k);

    std::string original = applyPermutation(encrypted, invPk);

    std::cout << "Дешифрованное слово: " << original << std::endl;

    return 0;

}

**Результат:**

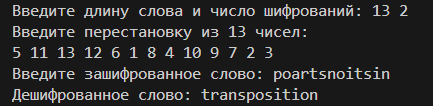
****

Рисунок 7

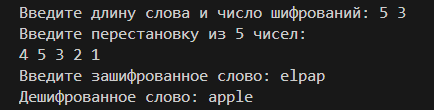
****

Рисунок 8

**Задача №5**

**Постановка задачи:**

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.

Формат входных данных:

В первой строке через пробел заданы высота h и ширина w матрицы (длина и ширина 10 или больше, но не больше 50, могут быть равны). В следующих строках заданы значения матрицы по строкам и столбцам. В матрице всегда есть только одна фигура. Фигура отстоит от краев матрицы минимум на один ноль. Начало координат в левом верхнем углу. Координаты растут вниз и вправо.

Формат выходных данных:

Координаты верхнего левого и правого нижнего угла прямоугольника отделенные пробелами. Координаты задаются номером строки и номером столбца. Нумерация начинается с 0.

**Ход решения:**

#include <iostream>

#include <vector>

int main() {

    int height;

    int width;

    std::cout << "Введите высоту матрицы: ";

    std::cin >> height;

    std::cout << "Введите ширину матрицы: ";

    std::cin >> width;

    std::vector<std::vector<int>> matrix(height, std::vector<int>(width));

    std::cout << "Введите матрицу построчно:" << std::endl;

    for (int i = 0; i < height; ++i) {

        for (int j = 0; j < width; ++j) {

            std::cin >> matrix[i][j];

        }

    }

    int min\_row = height;

    int max\_row = 0;

    int min\_col = width;

    int max\_col = 0;

    for (int i = 0; i < height; ++i) {

        for (int j = 0; j < width; ++j) {

            if (matrix[i][j] == 1) {

                if (i < min\_row) {

                    min\_row = i;

                }

                if (i > max\_row) {

                    max\_row = i;

                }

                if (j < min\_col) {

                    min\_col = j;

                }

                if (j > max\_col) {

                    max\_col = j;

                }

            }

        }

    }

    std::cout << "Координаты ограничивающего прямоугольника:" << std::endl;

    std::cout << min\_row - 1 << " " << min\_col - 1 << " " << max\_row + 1 << " " << max\_col + 1 << std::endl;

    return 0;

}

**Результат:**

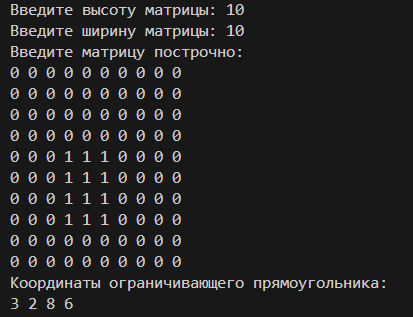


Рисунок 9

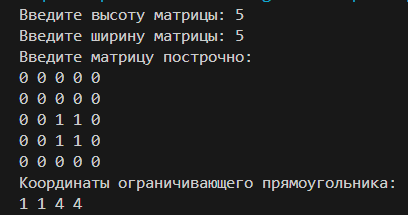


Рисунок 10

**Задача №6**

**Постановка задачи:**

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 и модуля управления мотором можно составить. Ваша программа должна ответить на n запросов.

Формат входных данных:

В первой строке пишем число n (не больше 50). Далее в n строках пишем по 4 натуральных числа (a, b, x, y).

Формат выходных данных:

Выводим n чисел через пробел, каждое число – максимальное число работающих пар из микроконтроллера и модуля управления мотором можно составить для строки.

**Ход решения:**

#include <iostream>

#include <algorithm>

#include <vector>

int main() {

    int n = 0;

    std::cout << "Введите количество запросов: \n";

    std::cin >> n;

    int a = 0;

    int b = 0;

    int x = 0;

    int y = 0;

    int usedB = 0;

    int remainingX = 0;

    int availableForA = 0;

    int usedA = 0;

    int result = 0;

    std::vector<int> vec\_result(n);

    int j = 0;

    std::cout << "Введите запросы: \n" << std::endl;

    for (int i = 0; i < n; ++i) {

        std::cin >> a >> b >> x >> y;

        usedB = std::min(b, x);

        remainingX = x - usedB;

        availableForA = remainingX + y;

        usedA = std::min(a, availableForA);

        result = usedA + usedB;

        vec\_result[j++] = result;

    }

    std::cout << "Результаты: \n";

    for (int i = 0; i < n; i++) {

        std::cout << vec\_result[i] << " ";

    }

    std::cout << std::endl;

    return 0;

}

**Результат:**

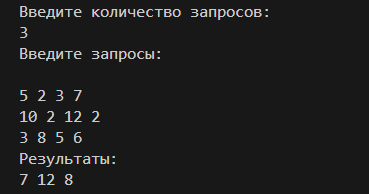
****

Рисунок 11

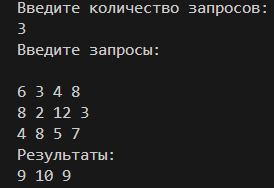
****

Рисунок 12

**Задача №7**

**Постановка задачи:**

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.

Формат входных данных:

В единственной строке находится строка из шести символов, состоящая из цифр и заглавных латинских букв.

Формат выходных данных:

Если строка является автомобильным номером, то необходимо вывести "Yes", в ином случае – "No" без кавычек.

**Ход решения:**

#include <iostream>

#include <cctype>

#include <string>

int main() {

    std::string input;

    std::cout << "Введите строку из шести символов: \n";

    std::cin >> input;

    bool isValid = true;

    if (input.length() != 6) {

        isValid = false;

    }

    if (isValid) {

        if (!std::isalpha(input[0]) || !std::isupper(input[0])) {

            isValid = false;

        }

        for (int i = 1; i <= 3; i++) {

            if (!std::isdigit(input[i])) {

                isValid = false;

            }

        }

        for (int i = 4; i <= 5; i++) {

            if (!std::isalpha(input[i]) || !std::isupper(input[i])) {

                isValid = false;

            }

        }

    }

    std::cout << "Результат: \n";

    if (isValid) {

        std::cout << "Yes" << std::endl;

    } else {

        std::cout << "No" << std::endl;

    }

    return 0;

}

**Результат:**

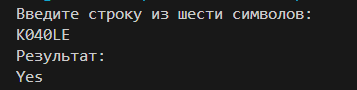
****

Рисунок 13

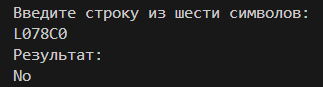


Рисунок 14

**Задание №8**

**Постановка задачи:**

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

**Ход решения:**

#include <Adafruit\_NeoPixel.h>

#define PIN 6

#define LED\_COUNT 64

Adafruit\_NeoPixel leds(LED\_COUNT, PIN, NEO\_GRB + NEO\_KHZ800);

int getIndex(int x, int y) {

  return (y % 2 == 0) ? (y \* 8 + x) : (y \* 8 + (7 - x));

}

void setup() {

  leds.begin();

}

void loop() {

  smiley();

  heart();

  fillEffect();

}

void smiley() {

  byte smile[8][8] = {

    {0,0,1,1,1,1,0,0},

    {0,1,0,0,0,0,1,0},

    {1,0,1,0,0,1,0,1},

    {1,0,0,0,0,0,0,1},

    {1,0,1,0,0,1,0,1},

    {1,0,0,1,1,0,0,1},

    {0,1,0,0,0,0,1,0},

    {0,0,1,1,1,1,0,0}

  };

  for (int i = 0; i < 3; i++) {

    leds.clear();

    for (int y = 0; y < 8; y++) {

      for (int x = 0; x < 8; x++) {

        if (smile[y][x]) {

          leds.setPixelColor(getIndex(x, y), leds.Color(255, 255, 0));

        }

      }

    }

    leds.show();

    delay(500);

    leds.clear();

    leds.show();

    delay(300);

  }

}

void heart() {

  byte heart[8][8] = {

    {0,1,1,0,0,1,1,0},

    {1,1,1,1,1,1,1,1},

    {1,1,1,1,1,1,1,1},

    {1,1,1,1,1,1,1,1},

    {0,1,1,1,1,1,1,0},

    {0,0,1,1,1,1,0,0},

    {0,0,0,1,1,0,0,0},

    {0,0,0,0,0,0,0,0}

  };

  for (int i = 0; i < 3; i++) {

    leds.clear();

    for (int y = 0; y < 8; y++) {

      for (int x = 0; x < 8; x++) {

        if (heart[y][x]) {

          leds.setPixelColor(getIndex(x, y), leds.Color(255, 0, 0));

        }

      }

    }

    leds.show();

    delay(500);

    leds.clear();

    leds.show();

    delay(300);

  }

}

void fillEffect() {

  int colors[3][3] = {

    {255, 0, 0},

    {0, 255, 0},

    {0, 0, 255}

  };

  for (int c = 0; c < 3; c++) {

    for (int i = 0; i < 64; i++) {

      leds.setPixelColor(i, leds.Color(colors[c][0], colors[c][1], colors[c][2]));

      leds.show();

      delay(30);

    }

    delay(300);

    leds.clear();

    leds.show();

    delay(300);

  }

}

**Результат:** [**https://wokwi.com/projects/434269731561778177**](https://wokwi.com/projects/434269731561778177)

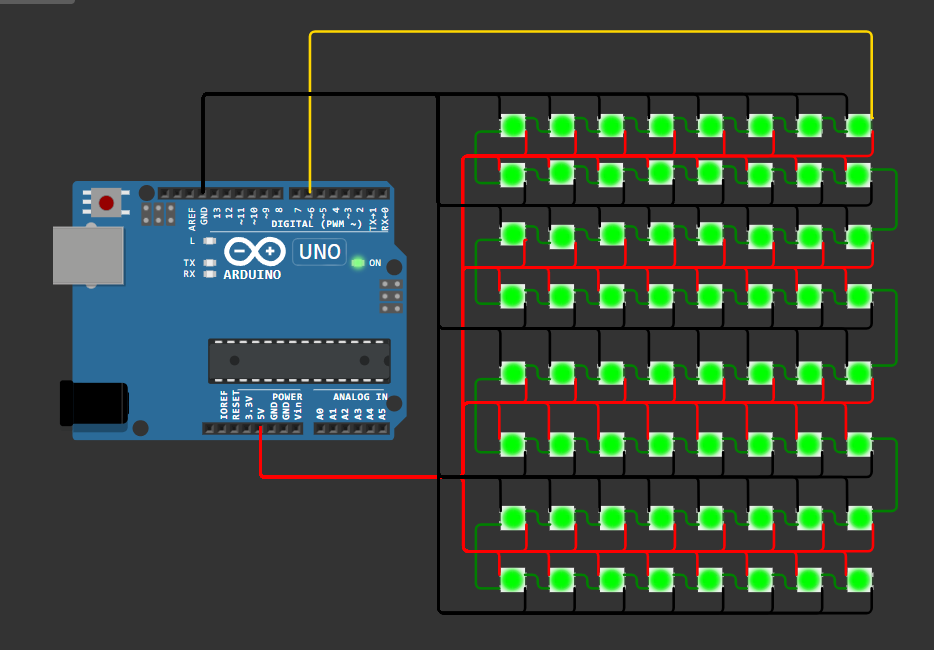
****

Рисунок 15

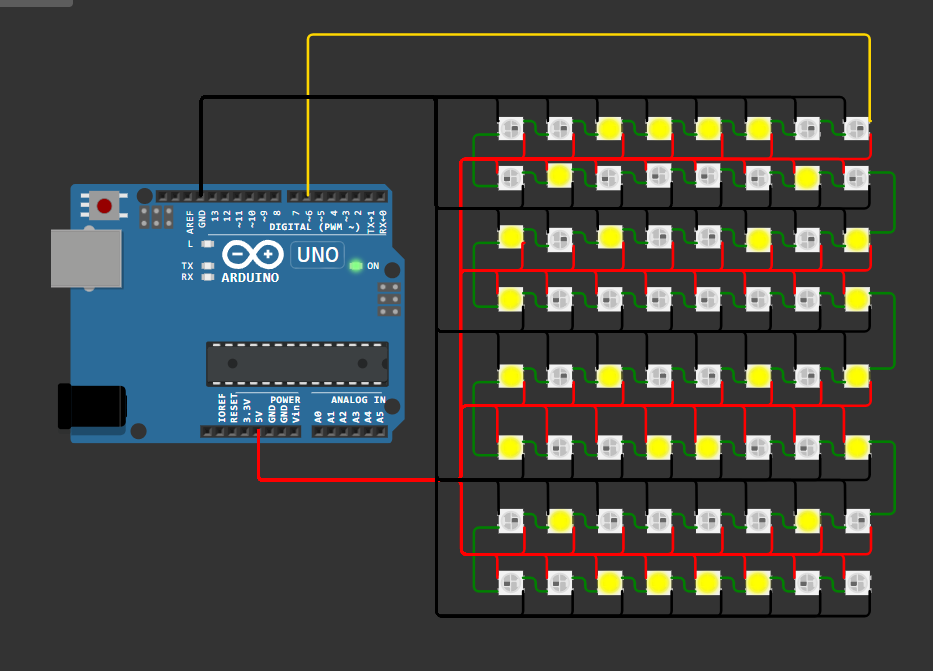


Рисунок 16

**Задание №9**

**Постановка задачи:**

Требования:

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.

Изменение значений фоторезистора осуществляется при помощи ползунка, изменение значения фоторезистора доступно только, когда запущен процесс моделирования.

Логика работы программы:

1. По умолчанию горит светодиод, имитирующий закрытую дверь
2. Микроконтроллер считывает значение фоторезистора с аналогово пина
3. Если значение на пине превышает 512, на определённое время загорается светодиод, имитирующий открытую дверь, в последовательный порт выводится сообщение о событии.
4. После истечения заданного временного промежутка проверяется значение фоторезистора, если оно всё ещё превышает 512, дверь должна остаться открытой, в противном случае нужно включить индикацию закрытой двери, в последовательный порт выводится сообщение о событии.

**Ход решения:**

#define PHOTO\_SENSOR A0

#define GREEN\_LED 5

#define RED\_LED 12

const int threshold = 512;

const unsigned long doorOpenTime = 3000;

void setup() {

  pinMode(GREEN\_LED, OUTPUT);

  pinMode(RED\_LED, OUTPUT);

  Serial.begin(9600);

  digitalWrite(RED\_LED, HIGH);

  digitalWrite(GREEN\_LED, LOW);

  Serial.println("Двери закрыты");

}

void loop() {

  int sensorValue = analogRead(PHOTO\_SENSOR);

  bool isDoorOpen = false;

  if (digitalRead(GREEN\_LED) == HIGH) {

    isDoorOpen = true;

  }

  if (sensorValue > threshold) {

    digitalWrite(RED\_LED, LOW);

    digitalWrite(GREEN\_LED, HIGH);

    if (!isDoorOpen) {

      Serial.println("Двери открываются!");

    }

    delay(doorOpenTime);

    sensorValue = analogRead(PHOTO\_SENSOR);

    if (sensorValue <= threshold) {

      digitalWrite(GREEN\_LED, LOW);

      digitalWrite(RED\_LED, HIGH);

      Serial.println("Двери закрываются");

    }

  }

}

**Результат:** [**https://wokwi.com/projects/434446829419518977**](https://wokwi.com/projects/434446829419518977)

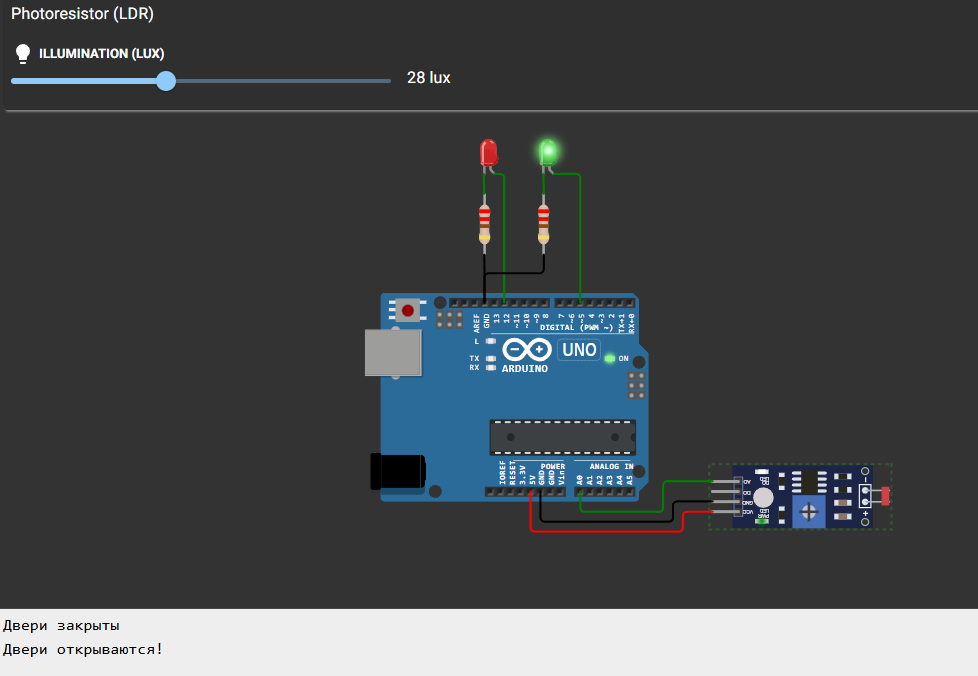
****

Рисунок 17

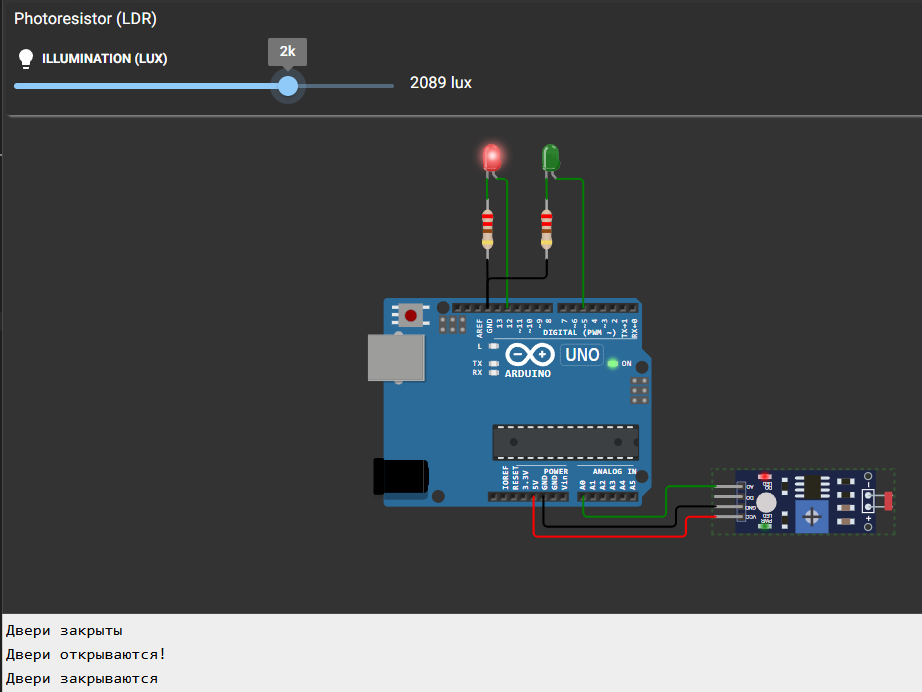


Рисунок 18

**Задание №10**

**Постановка задачи:**

Требования:

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

Логика работы программы:

1. Программа находится в ожидании ввода данных в последовательный порт
2. Когда в последовательный порт вводятся данные, программа проверяет их на корректность (Должны приниматься только числовые значения в диапазоне от 0 до 180)
3. Если введён угол поворота, отличный от текущего угла, сервопривод плавно поворачивается в заданное положение.

**Ход решения:**

#include <Servo.h>

Servo myServo;

int currentAngle = 90;

void setup() {

  Serial.begin(9600);

  myServo.attach(4);

  myServo.write(currentAngle);

  Serial.println("Введите угол (0-180):");

}

void loop() {

  if (Serial.available() > 0) {

    int newAngle = Serial.parseInt();

    if (newAngle >= 0 && newAngle <= 180) {

      if (newAngle != currentAngle) {

        Serial.print("Сервопривод поворачивается на: ");

        Serial.println(newAngle);

        if (newAngle > currentAngle) {

          for (int i = currentAngle; i <= newAngle; i++) {

            myServo.write(i);

            delay(15);

          }

        } else {

          for (int i = currentAngle; i >= newAngle; i--) {

            myServo.write(i);

            delay(15);

          }

        }

        currentAngle = newAngle;

      }

    } else {

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

    }

  }

}

**Результат:** [**https://wokwi.com/projects/434456265361517569**](https://wokwi.com/projects/434456265361517569)

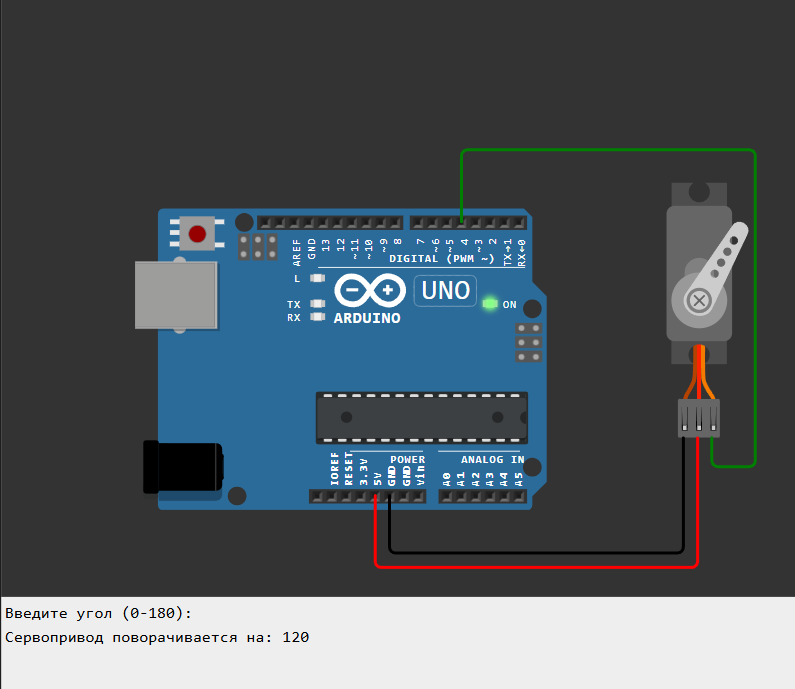
****

Рисунок 19

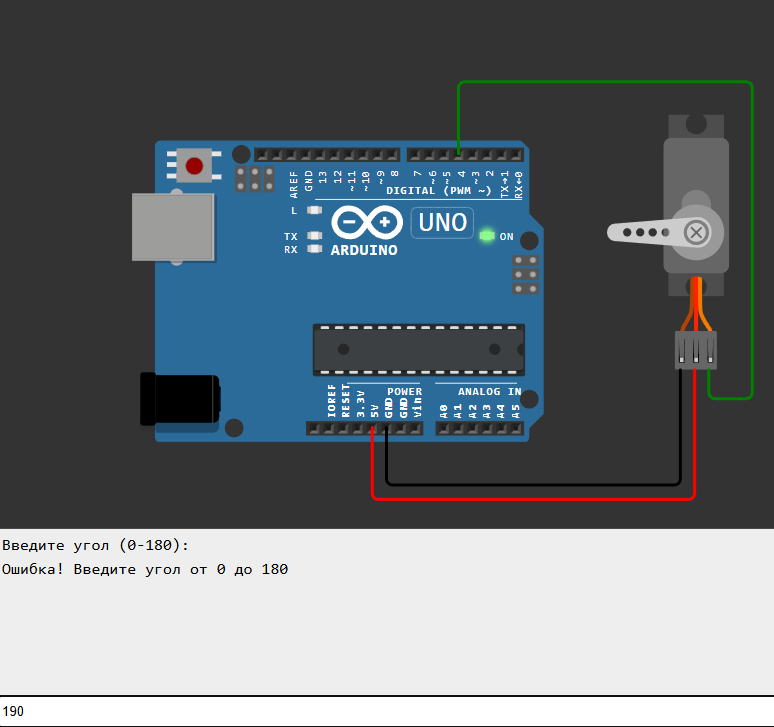
****

Рисунок 20

Задание №11

**Постановка задачи:**

Вариант 19 (по списку). Найдите самый большой объект на изображении. Определите центр самого большого объекта. Обведите самый большой объект красной рамкой.

**Ход решения:**

import cv2  
import numpy as np  
  
image = cv2.imread("test3.jpg")  
if image is None:  
 print("Ошибка: изображение не загружено")  
 exit()  
  
cv2.imshow("Original", image)  
cv2.waitKey(0);  
  
blurred = cv2.GaussianBlur(image, (11, 11), 0)  
cv2.imshow("Blurred image", blurred)  
cv2.waitKey(0)  
  
hsv = cv2.cvtColor(blurred, cv2.COLOR\_BGR2HSV)  
cv2.imshow("HSV", hsv)  
cv2.waitKey(0)  
  
lower\_bound = np.array([0, 50, 50])  
upper\_bound = np.array([179, 255, 255])  
  
mask = cv2.inRange(hsv, lower\_bound, upper\_bound)  
cv2.imshow("Mask", mask)  
cv2.waitKey(0)  
  
contours, \_ = cv2.findContours(mask, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)  
  
if not contours:  
 print("Контуры не найдены")  
else:  
 largest\_contour = max(contours, key=cv2.contourArea)  
  
 x, y, w, h = cv2.boundingRect(largest\_contour)  
  
 center\_x = x + w // 2  
 center\_y = y + h // 2  
  
 cv2.rectangle(image, (x, y), (x + w, y + h), (0, 0, 255), 3)  
  
 cv2.circle(image, (center\_x, center\_y), 10, (0, 0, 255), -1)  
  
 cv2.imshow("Largest Object", image)  
 cv2.waitKey(0)  
 cv2.destroyAllWindows()

**Результат:**

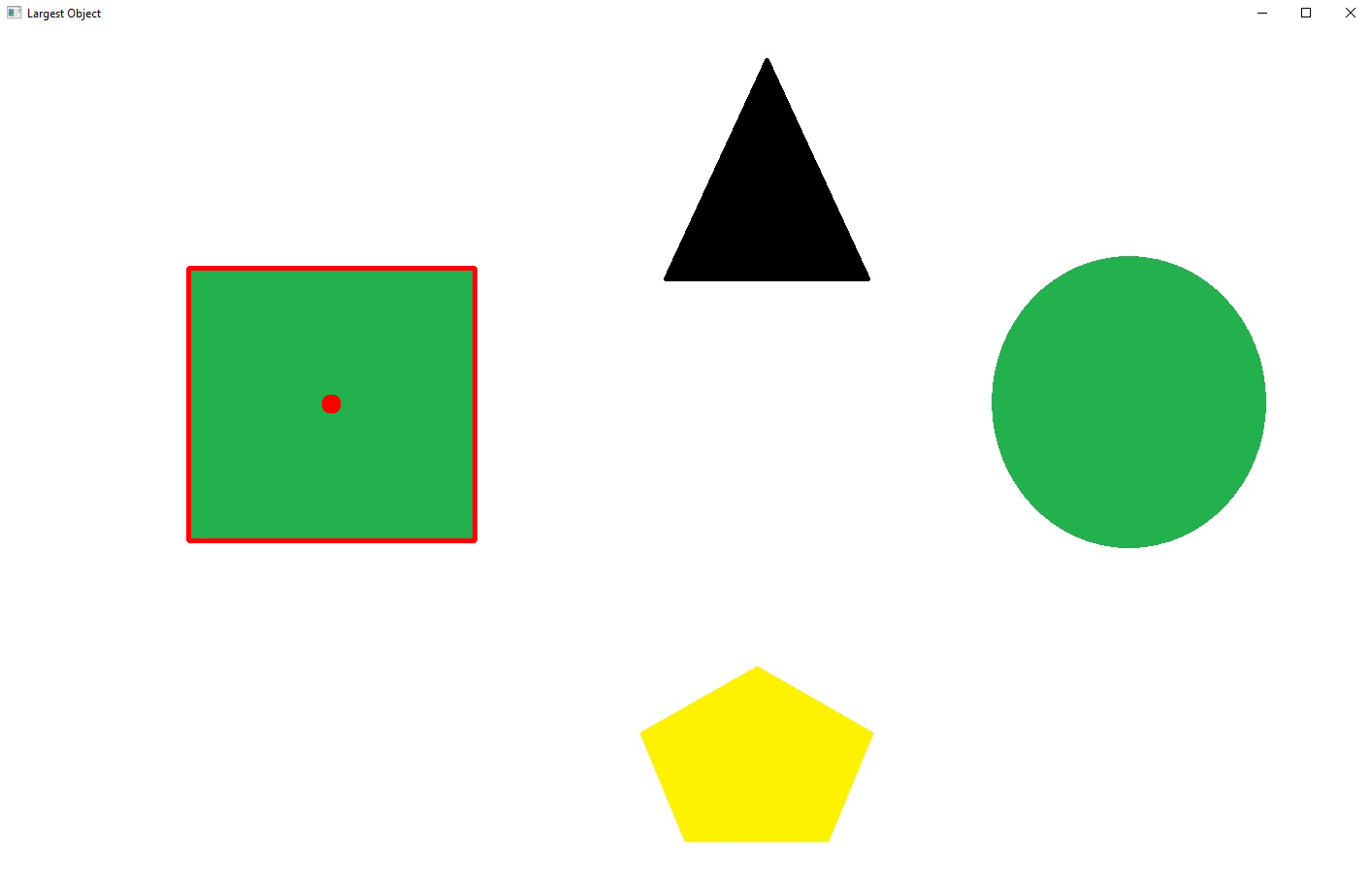
****

Рисунок 21

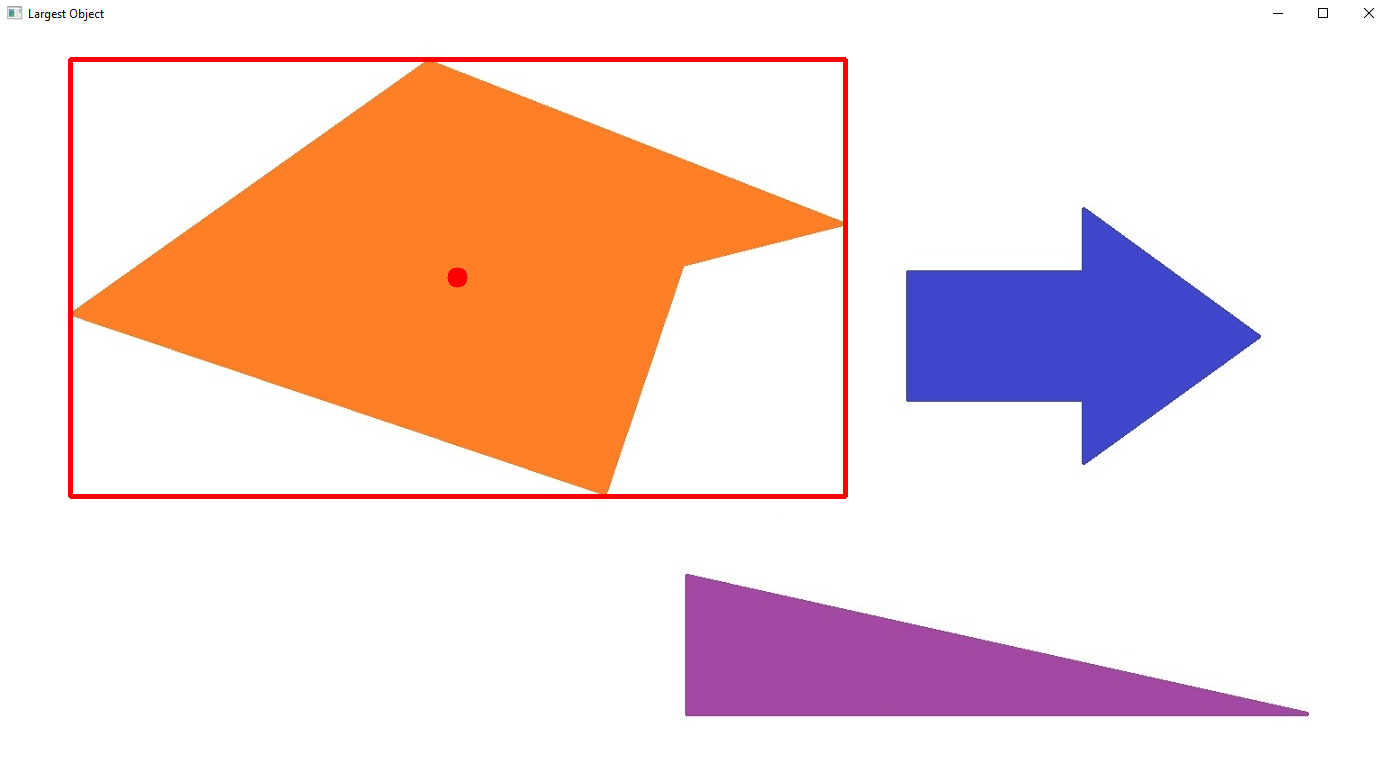
****

Рисунок 22

**Отчёт по посещению экскурсии 22.06.2025 в АО «СО ЕЭС»**

22 июня 2025 года наша группа посетила филиал АО «Системный оператор Единой энергетической системы» (СО ЕЭС) Иркутского регионального диспетчерского управления, расположенный по адресу: ул. Ширямова 54а. Нас провели по ряду диспетчерских залов, подробно рассказали о ключевых элементах функционирования энергосистемы региона. Особое внимание было уделено демонстрации диспетчерских панелей и диаграмм, наглядно отображающих напряжение, загрузку сетей и потребление электроэнергии в Иркутской области. В целом, экскурсия оказалась познавательной, хотя по моей специальности работать здесь не особенно хотелось, но для прохождения практики вариант вполне подходящий.

АО «СО ЕЭС» осуществляет централизованное оперативно‑диспетчерское управление Единой энергетической системой России. Основные задачи компании включают:

* Прогнозирование потребления электроэнергии и мощности
* Планирование и оперативное управление режимами работы энергосистемы
* Обеспечение круглосуточного контроля за энергообъектами и взаимодействие с электроэнергетическими субъектами
* Внедрение дистанционного управления объектами генерации и линиями электропередач
* Развитие Единой информационной модели ЕЭС России (ЕИМ) и IT‑инфраструктуры

IT‑подразделение АО «СО ЕЭС» отвечает за:

* Надёжность информационных систем в диспетчерских центрах
* Автоматизацию технологических процессов и телекоммуникации
* Поддержку структуры диспетчерских центров, IT‑активов и документированию процессов

Экскурсия впечатлила: было интересно увидеть, как в реальном времени отслеживается и управляется энергосистема региона. Особенно запомнились диспетчерские залы с большими экранами, на которых отображаются схемы нагрузки и передачи. Несмотря на любопытство, я понял, что профессионально это направление мне не подходит — оно более техническое и специализированное, чем моя будущая специальность. Но участникам практики, особенно интересующимся энергетикой или IT‑инфраструктурой, такие объекты — отличное место для набора практического опыта. На будущее, возможно, рассмотрю вариант практики здесь.

Экскурсия в АО «СО ЕЭС» в Иркутске дала ценный практический опыт. Полученные знания о диспетчерском управлении и цифровой инфраструктуре региона расширили представление о том, как устроена энергетика на уровне территории. Хотя дальнейшая работа здесь не кажется мне оптимальным выбором по профилю, участие в практических проектах АО «СО ЕЭС» может быть полезно для студентор, интересующихся энергетикой, IT и киберзащитой.

**Отчёт по посещению экскурсии 24.06.2025 в АО «ИСПсистем»**

22 июня 2025 года мы посетили иркутский офис компании ISPsystem в рамках учебной практики. Экскурсия началась с вводной презентации, где представители компании рассказали, чем занимается ISPsystem, как она развивалась и какие направления сейчас активно развиваются.

Затем нам подробно рассказали о внутренних структурах компании: какие есть отделы, какие специалисты в них работают и чем конкретно занимаются. Мы узнали о таких направлениях, как frontend и backend разработка, тестирование (QA), UI/UX-дизайн, техническая поддержка (1-я и 2-я линии), менеджмент, аналитика, сопровождение продуктов и многое другое. Было подчёркнуто, что в компании востребованы не только программисты, но и специалисты других профилей — поэтому работать здесь может тот, кому интересно развитие в IT даже без глубоких знаний языков программирования. Нам показали производственные и административные помещения, включая замечательно оборудованную кухню, комнату отдыха и рабочие зоны. Всё выполнено с заботой о сотрудниках — комфортно и продуманно.

Мы узнали, что ISPsystem — компания‑разработчик платформ для интегрированного управления IT‑инфраструктурой, работу ведёт с 2004 года. Обслуживает свыше 200 000 серверов более чем в 100 странах, сотрудничает с тысячами компаний.

Их основные продукты:

* VMmanager — автоматизация управления виртуальными серверами (VPS/VDS)
* DCImanager — управление выделенными серверами, шкафами, оборудованием и учёт ресурсов
* BILLmanager — система поддержки продаж и биллинга
* DNSmanager, IPmanager — контроль DNS и IP‑адресов

Платформы отличаются гибкостью, масштабируемостью, простым UI, наличием документации и открытым API.

Экскурсия произвела сильное впечатление: почувствовал, что работа в компании почти идеально совпадает с моим профилем и интересами. Позитивные, дружелюбные сотрудники и комфортные условия вызывают желание проходить здесь практику, а впоследствии — даже работать.

Посещение ISPsystem дало мне не только представление о работе современной IT-компании, но и помогло определиться с вектором развития. Компания впечатляет своим техническим уровнем, разнообразием продуктов и корпоративной культурой. Условия работы и коллектив — очень хорошие, поэтому хотелось бы пройти здесь практику и развиваться далее.