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 Воробьева Д. М., ИСИБ 24-1

(ФИО, группа, подпись)

Руководитель практики от института ИТиАД

Кононенко Роман Владимирович, доцент

(ФИО, должность, подпись)

https://www.superjob.ru/resume/programmist-razrabotchik-55731973.html

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

https://irkutsk.hh.ru/resume/35d23296ff0efcb53c0039ed1f4e4b764b6249

# Содержание

[**Отчёт** 1](#_Toc209739595)

[**Содержание** 2](#_Toc209739596)

[**Введение** 3](#_Toc209739597)

[**Задание** **на** **практику** **№1** 4](#_Toc209739598)

[**Задание** **на практику № 2** 8](#_Toc209739599)

[**Задание** **на практику № 3** 12](#_Toc209739600)

[**Задание** **на практику № 4** 16](#_Toc209739601)

[**Задание** **на практику № 5** 19](#_Toc209739602)

[**Задание** **на практику № 6** 23](#_Toc209739603)

[**Задание** **на практику № 7** 25](#_Toc209739604)

[**Задание на практику № 8** 27](#_Toc209739605)

[**Задание на практику № 9** 31](#_Toc209739606)

[**Задание** **на практику № 10** 36](#_Toc209739607)

[**Задание на практику № 11** 39](#_Toc209739608)

[**Отчет** **по экскурсии по АО «СО ЕЭС» 22.06.2025** 41](#_Toc209739609)

[**Отчет** **по** **экскурсии** **в АО «ИСПсистем» 24.06.2025** 42](#_Toc209739610)

[**Заключение** 43](#_Toc209739611)

[**Список** **литературы** 44](#_Toc209739612)

# Введение

Современный этап развития общества характеризуется повсеместной цифровизацией и трансформацией всех сфер жизнедеятельности под влиянием информационных технологий. Программное обеспечение стало неотъемлемой частью бизнес-процессов, научных исследований, образования и повседневного общения. В этих условиях профессия программиста приобретает ключевое значение, требуя от специалиста не только глубоких теоретических знаний, но и уверенных практических навыков, а также понимания полного жизненного цикла разработки программного обеспечения.

Учебная практика является важнейшим элементом образовательной программы, связующим звеном между академическими знаниями, полученными в стенах университета, и реальными профессиональными задачами. Она позволяет применить теоретические основы алгоритмизации, языков программирования, баз данных и других дисциплин для решения конкретных практических проблем. Именно в ходе практики формируется представление о работе в коллективе, использовании современных инструментов разработки и методологиях управления.

# Задание на практику №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.



Рисунок 1 – пример движения

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)



Рисунок 2 – пример ввода-вывода

#include <iostream>

#include <vector>

#include <random>

#include <algorithm>

void Moon\_Generate(std::vector<std::vector<int>> &Moon, size\_t height) {

    std::random\_device rd;

    std::mt19937 gen(rd());

    int min = 1;

    int max = 100;

    std::uniform\_int\_distribution<> rand(min, max);

    Moon.resize(height);

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

        Moon[i].resize(i + 1);

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

            Moon[i][j] = rand(gen);

        }

    }

}

void find\_Path(std::vector<std::vector<int>> &Moon,

    std::vector<std::vector<int>> &Dp\_Moon, size\_t height, int &time, std::vector<int> &Path) {

    Dp\_Moon.resize(height);

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

        Dp\_Moon[i].resize(i + 3, 99999999);

    }

    Dp\_Moon[0][0] = Moon[0][0];

    for (size\_t i = 1; i < height; ++i) {

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

            int min\_prev = 9999;

            if (j > 0) min\_prev = std::min(min\_prev, Dp\_Moon[i-1][j-1]);

            if (j < i) min\_prev = std::min(min\_prev, Dp\_Moon[i-1][j]);

            if (j < i-1) min\_prev = std::min(min\_prev, Dp\_Moon[i-1][j+1]);

            Dp\_Moon[i][j] = Moon[i][j] + min\_prev;

        }

    }

    size\_t current = std::min\_element(Dp\_Moon[height - 1].begin(), Dp\_Moon[height - 1].end()) - Dp\_Moon[height - 1].begin();

    Path.push\_back(Moon[height-1][current]);

    time = Dp\_Moon[height - 1][current];

    for (int i = height-2; i >= 0; --i) {

        if (current == 0) {

            current = std::min\_element(Dp\_Moon[i].begin(), Dp\_Moon[i].begin() + 2) - Dp\_Moon[i].begin();

        } else {

            current = std::min\_element(Dp\_Moon[i].begin() + current - 1, Dp\_Moon[i].begin() + current + 1) - Dp\_Moon[i].begin();

        }

        Path.push\_back(Moon[i][current]);

    }

    std::reverse(Path.begin(), Path.end());

}

int main() {

    size\_t height;

    std::cout << "Enter the height of the pyramid: ";

    std::cin >> height;

    std::vector<std::vector<int>> Moon(height), Dp\_Moon(height);

    std::vector<int> Path;

    Moon\_Generate(Moon, height);

    int time;

    find\_Path(Moon, Dp\_Moon, height, time, Path);

    std::cout << "your Pyramid:\n";

    for (auto i : Moon) {

        for (auto j : i) {

            std::cout << j << " ";

        }

        std::cout << "\n";

    }

    std::cout << "Your Time: " << time << "\n";

    std::cout << "Your Path: ";

    for (auto j : Path) {

        std::cout << j << " ";

    }

    std::cout << "\n";

    return 0;

}

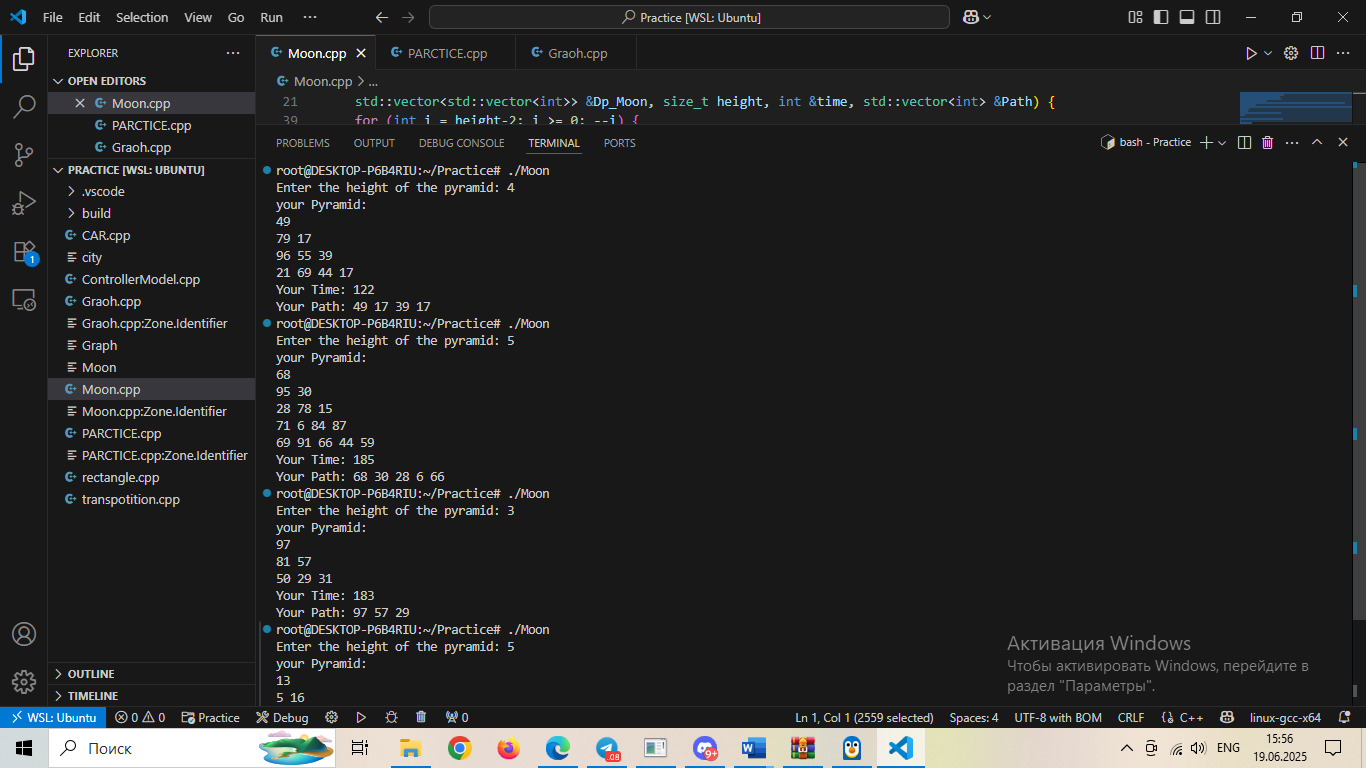


Рисунок 3 – Тесты.

# Задание на практику № 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.



Рисунок 4 – граф



Рисунок 5 – обозначение графа

**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)



Рисунок 6 – пример ввода-вывода

#include <iostream>

#include <vector>

enum Colors { White, Grey, Black };

class Graph {

private:

    std::vector<std::vector<int>> vertices\_;

    std::vector<Colors> colors\_;

public:

    explicit Graph(int n, int m) {

        vertices\_.resize(n);

        colors\_.resize(n, White);

        for (int i = 0; i < m; i++) {

            int v1 = 0;

            int v2 = 0;

            std::cin >> v1 >> v2;

            --v1;

            --v2;

            vertices\_[v1].push\_back(v2);

            vertices\_[v2].push\_back(v1);

        }

    }

    void DFS(int index) {

        colors\_[index] = Grey;

        for (int ner : vertices\_[index]) {

            if (colors\_[ner] == White) {

                DFS(ner);

            }

        }

        colors\_[index] = Black;

    }

    void ComponentCount() {

        int count = 0;

        for (int i = 0; i < static\_cast<int>(vertices\_.size()); i++) {

            if (colors\_[i] == White) {

                DFS(i);

                count++;

            }

        }

        std::cout << "You have " << count - 1 << " Components\n";

    }

};

int main() {

    int n = 0, m = 0;

    std::cout << "write your number of vertices: ";

    std::cin >> n;

    std::cout << "write your number of edges: ";

    std::cin >> m;

    std::cout << "write your edges:\n";

    Graph graph(n, m);

    graph.ComponentCount();

    return 0;

}

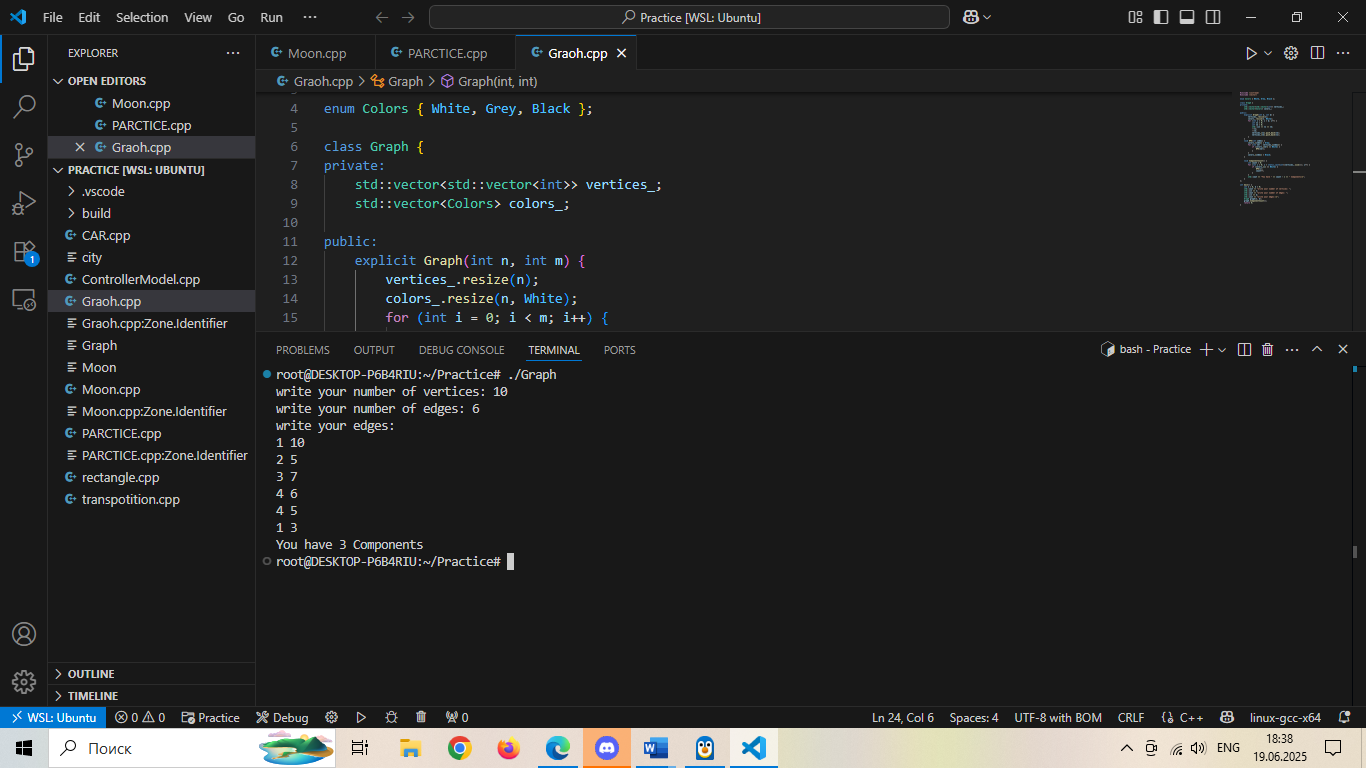


Рисунок 7 – тест 1

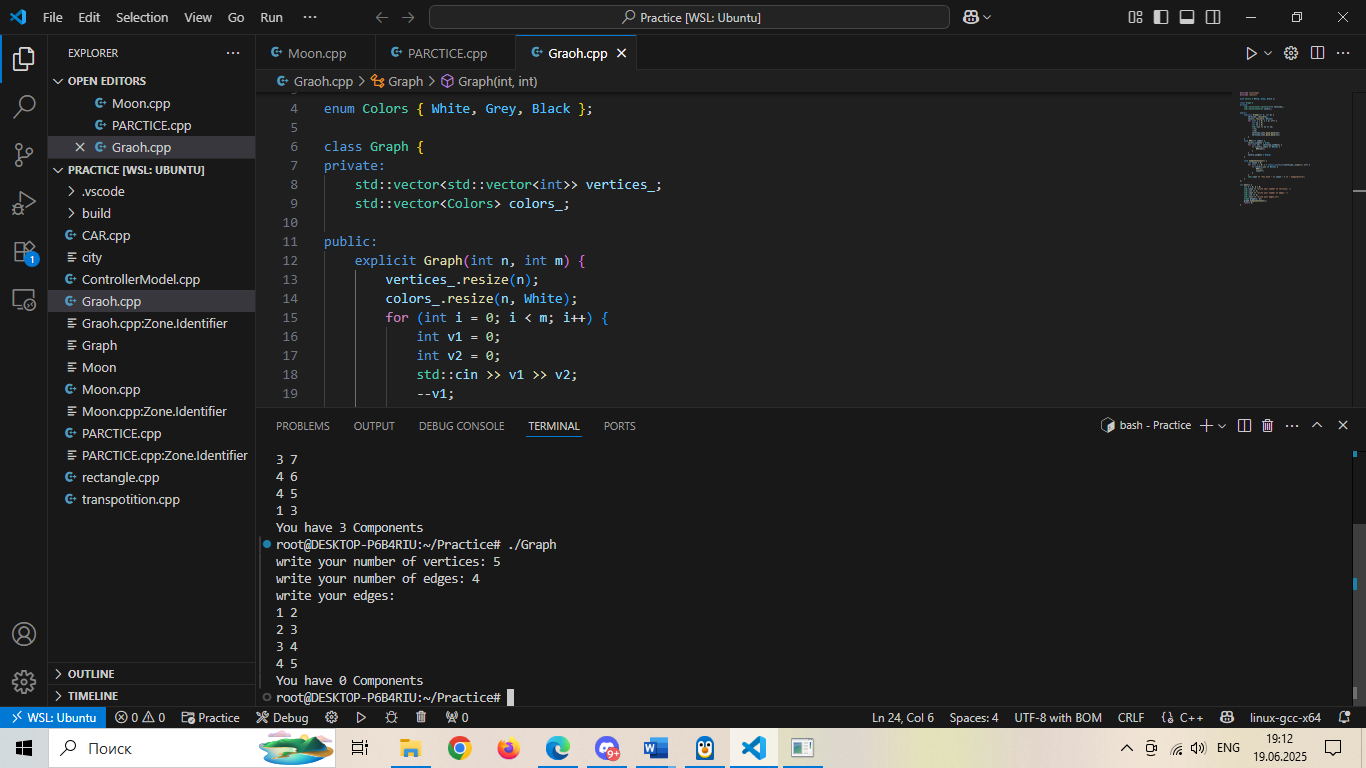


Рисунок 8 – тест 2

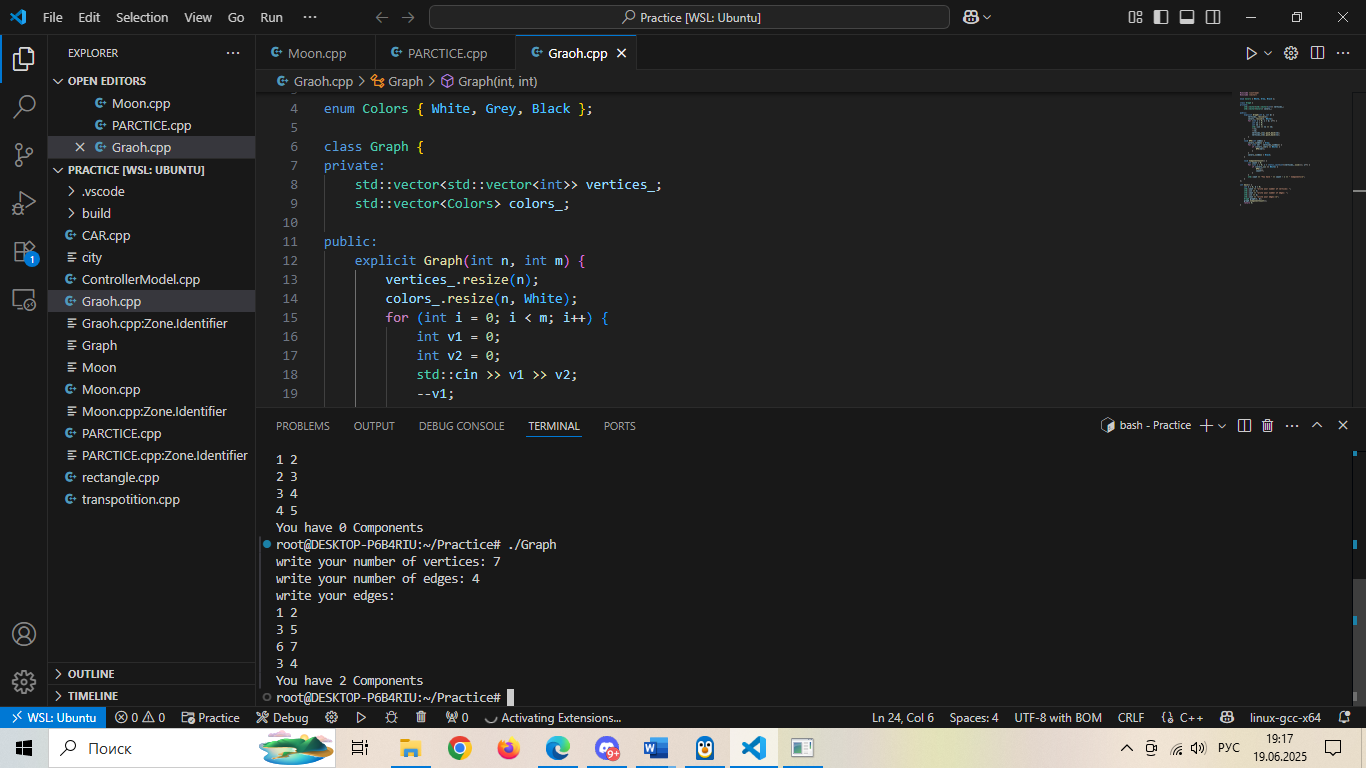


Рисунок 9 – тест 3

# Задание на практику № 3

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.

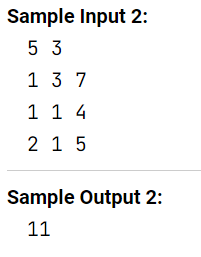


Рисунок 9 – пример ввода-вывода

#include <iostream>

#include <vector>

class fanfic {

private:

    std::vector<int> city;

    int size;

public:

    fanfic(int n) : size(n + 2), city(n + 2, 0) {}

    void modify(int i, int d) {

        while (i < size) {

            city[i] += d;

            i = i | (i + 1);

        }

    }

    void set(int i, int x) {

        int current = sum(i) - sum(i - 1);

        int d = x - current;

        modify(i, d);

    }

    int sum(int i) {

        int res = 0;

        while (i > 0) {

            res += city[i];

            i = (i & (i + 1)) - 1;

        }

        return res;

    }

    int citysum(int l, int r) {

        return sum(r) - sum(l - 1);

    }

};

int main() {

    int n, m;

    std::cout << "write number of streets and number of calls: ";

    std::cin >> n >> m;

    std::cout << "write '1 <number> <snow\_high>' to change the amount of snow on the street\n";

    std::cout << "write '2 left\_street right\_street' to see the amount of snow between those streets\n";

    fanfic city(n);

    for (int i = 0; i < m; i++) {

        int a, b, c;

        std::cin >> a >> b >> c;

        if (a == 1) {

            city.set(b, c);

            std::cout << "Snow number succesfuly changed\n";

        } else if (a == 2) {

            std::cout << "The sum of snow between streets " << b << " and " << c

                      << " is: " << city.citysum(b, c) << "\n";

        } else {

            std::cout << "wrong command\n";

        }

    }

    return 0;

}

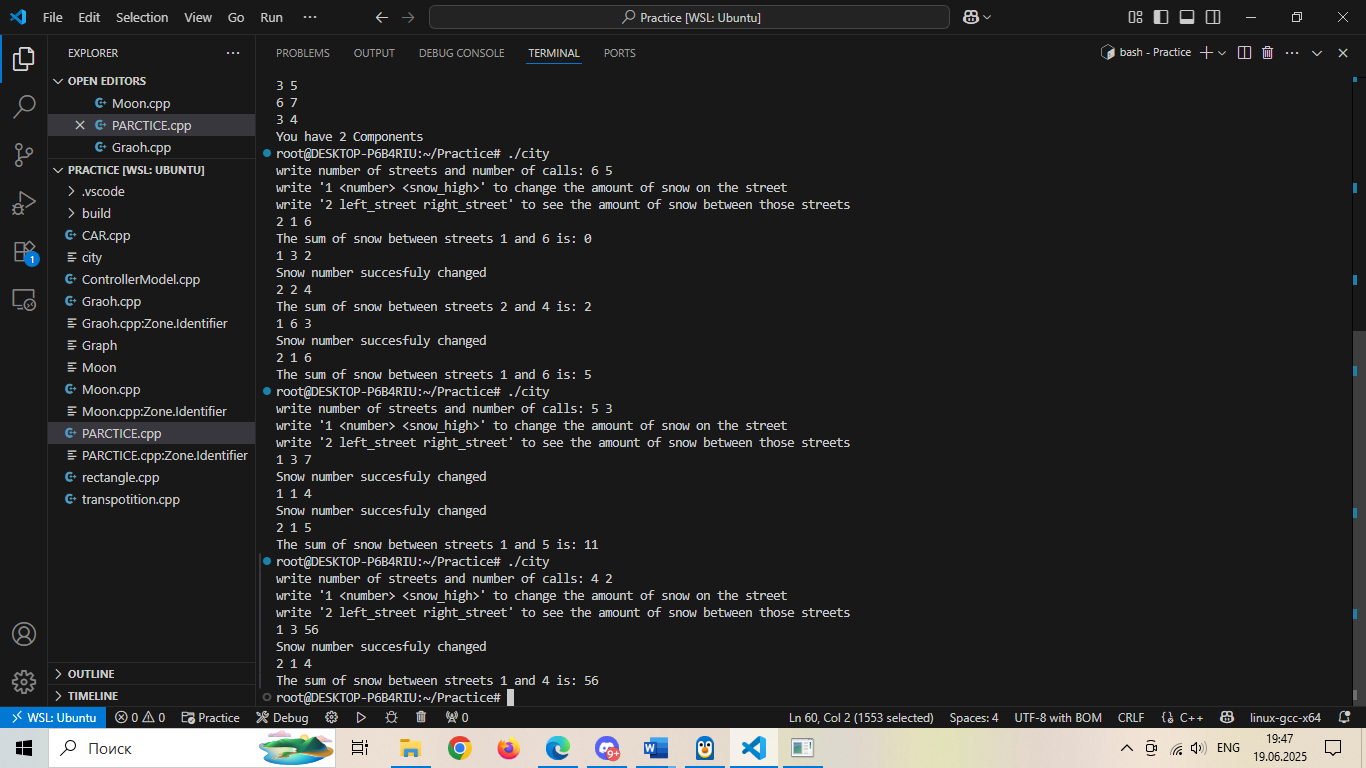


Рисунок 10 – тесты

# Задание на практику № 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. 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.

 - -

Рисунок 11 – пример перестановки

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.



Рисунок 12 – пример ввода-вывода

#include <iostream>

#include <vector>

#include <string>

int main()

{

    int n, k;

    std::cout << "Enter lenght of the word and number of transposition:";

    std::cin >> n >> k;

    std::vector<int> partition(n), ans(n, 0);

    std::string word;

    std::cout << "Enter transposition code:\n";

    for (int i = 0; i < n; i++) {

        std::cin >> partition[i];

        partition[i]--;

    }

    std::cout << "Enter transposited word:\n";

    std::cin >> word;

    for (int i = 0; i < n; i++) {

        int num = i;

        for (int j = 0; j < k; j++) {

            num = partition[num];

        }

        ans[num] = i;

    }

    std::cout << "Your restored word:\n";

    for (int i = 0; i < n; i++) {

        std::cout << word[ans[i]];

    }

    std::cout << "\n";

}

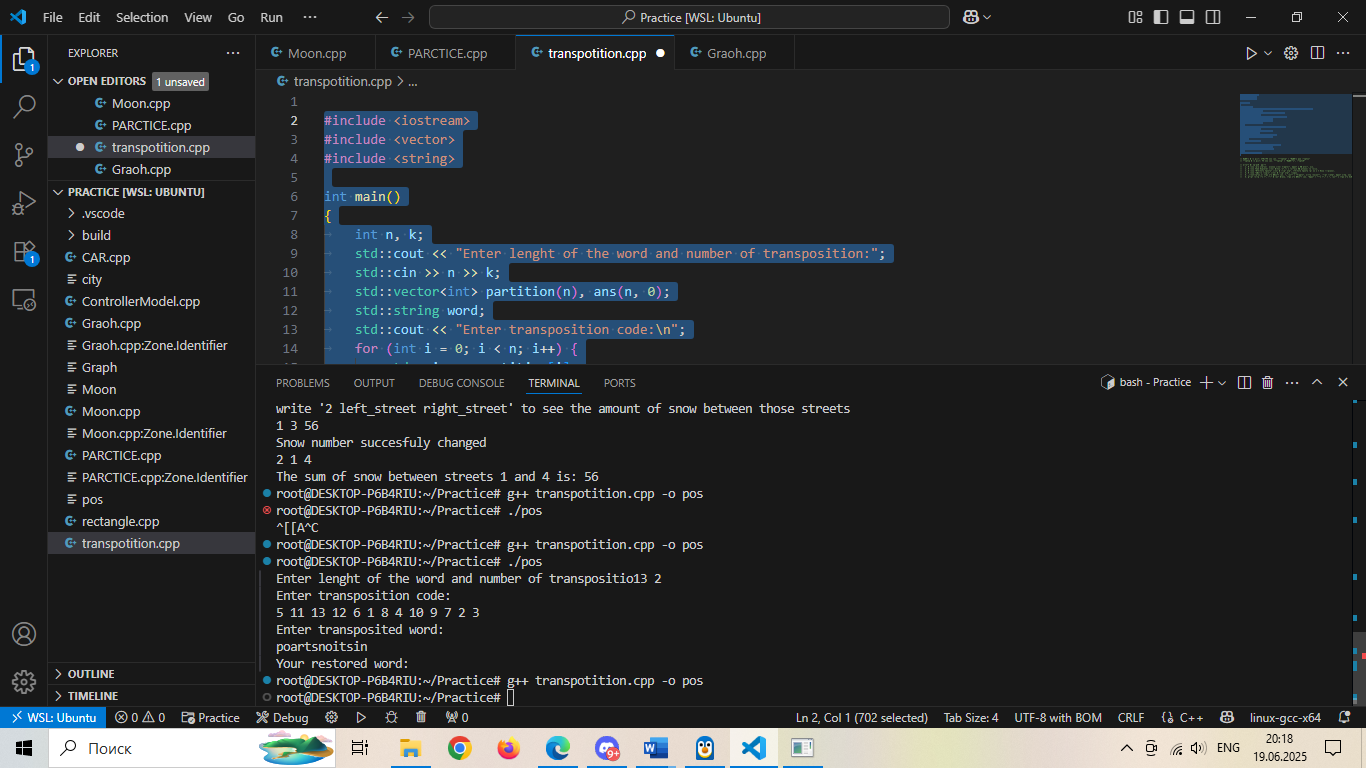


Рисунок 13 - тест

# Задание на практику № 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.

**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.



Рисунок 14 - пример ввода-вывода

#include <iostream>

#include <vector>

#include <string>

int main()

{

    int n, m;

    std::cout << "Enter widht and lenght of rectange:";

    std::cin >> n >> m;

    std::vector<std::vector<int>> matrix(n);

    std::pair<size\_t, size\_t> first, second;

    bool flag = true;

    std::cout << "Enter rectange:\n";

    for (int i = 0; i < n; i++) {

        matrix[i].resize(m);

        for (int j = 0; j < m; j++) {

            std::cin >> matrix[i][j];

            if (matrix[i][j] == 1) {

                if (flag) {

                    first.first = i - 1;

                    first.second = j - 1;

                    flag = false;

                } else {

                second.first = i + 1;

                second.second = j + 1;

                }

            }

        }

    }

    std::cout << "First point: " << first.first << " " << first.second << " second point: " << second.first << " " << second.second << "\n";

}

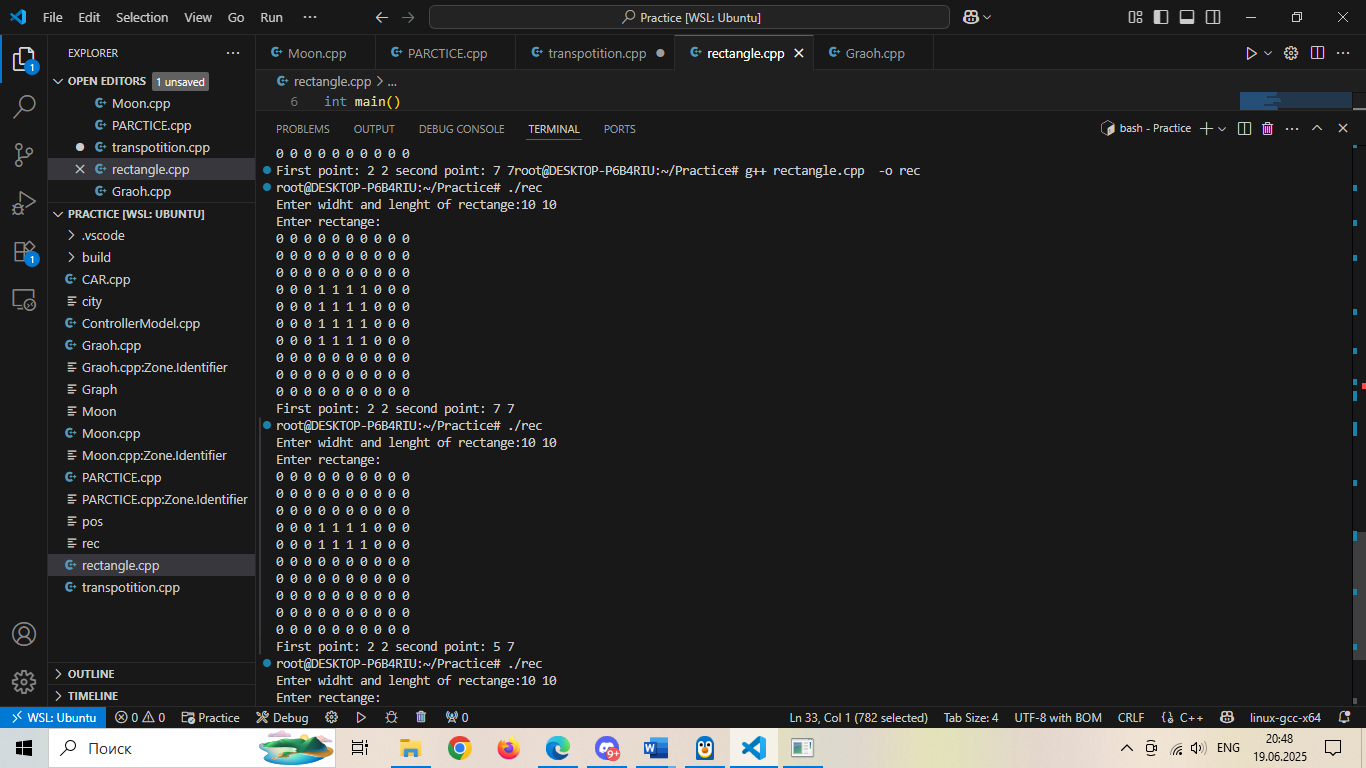


Рисунок 14 – тесты 1, 2

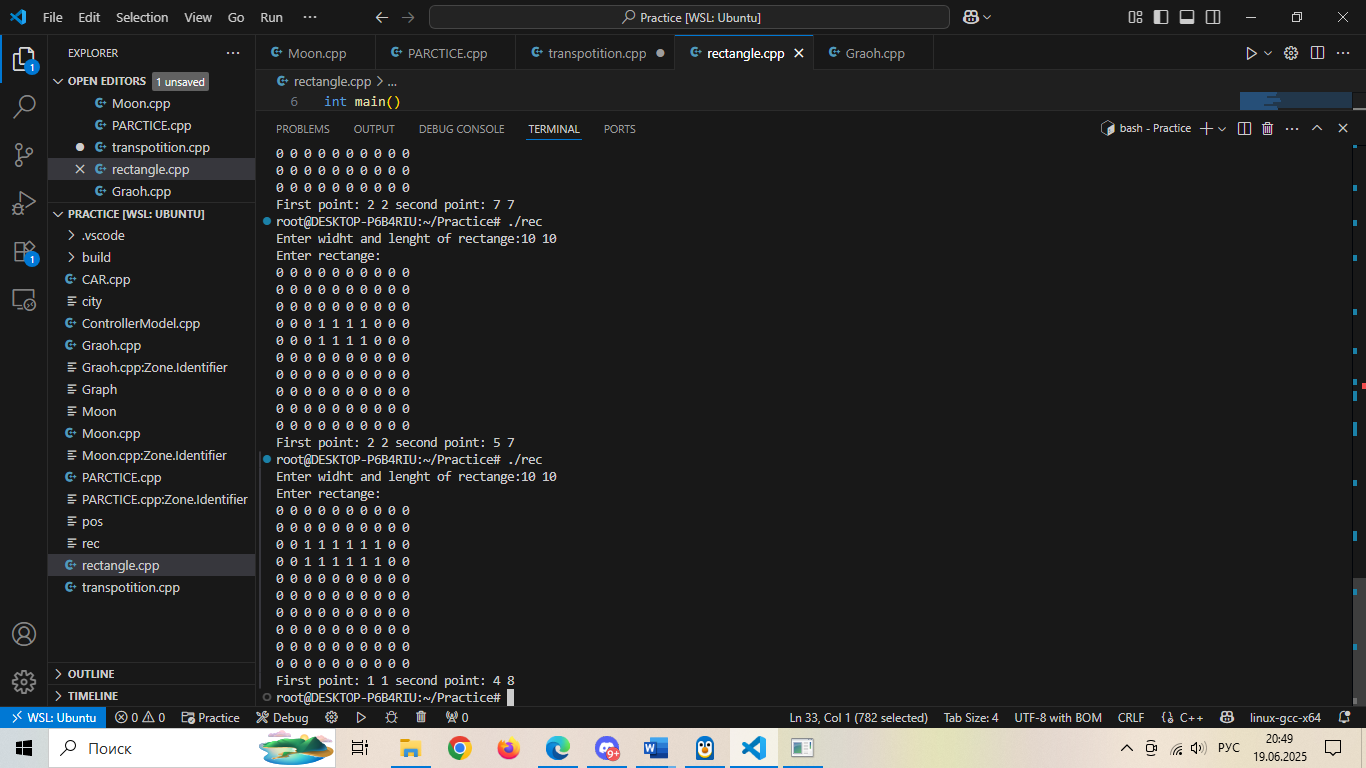


Рисунок 15 – тест 3

# Задание на практику № 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 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.



Рисунок 16 – пример ввода-вывода

#include <iostream>

#include <vector>

#include <string>

int main()

{

    int n;

    std::cout << "Enter the number of calls: ";

    std::cin >> n;

    int a, b, x, y;

    std::vector<int> ans(n);

    for (int i = 0; i < n; i++) {

        std::cout << "Enter the number A, B, x, y: ";

        std::cin >> a >> b >> x >> y;

        ans[i] = std::min(b, x) + std::min(a, std::max(0, x - b) + y);

        std::cout << "The max number of Plates:" << ans[i] << "\n";

    }

}

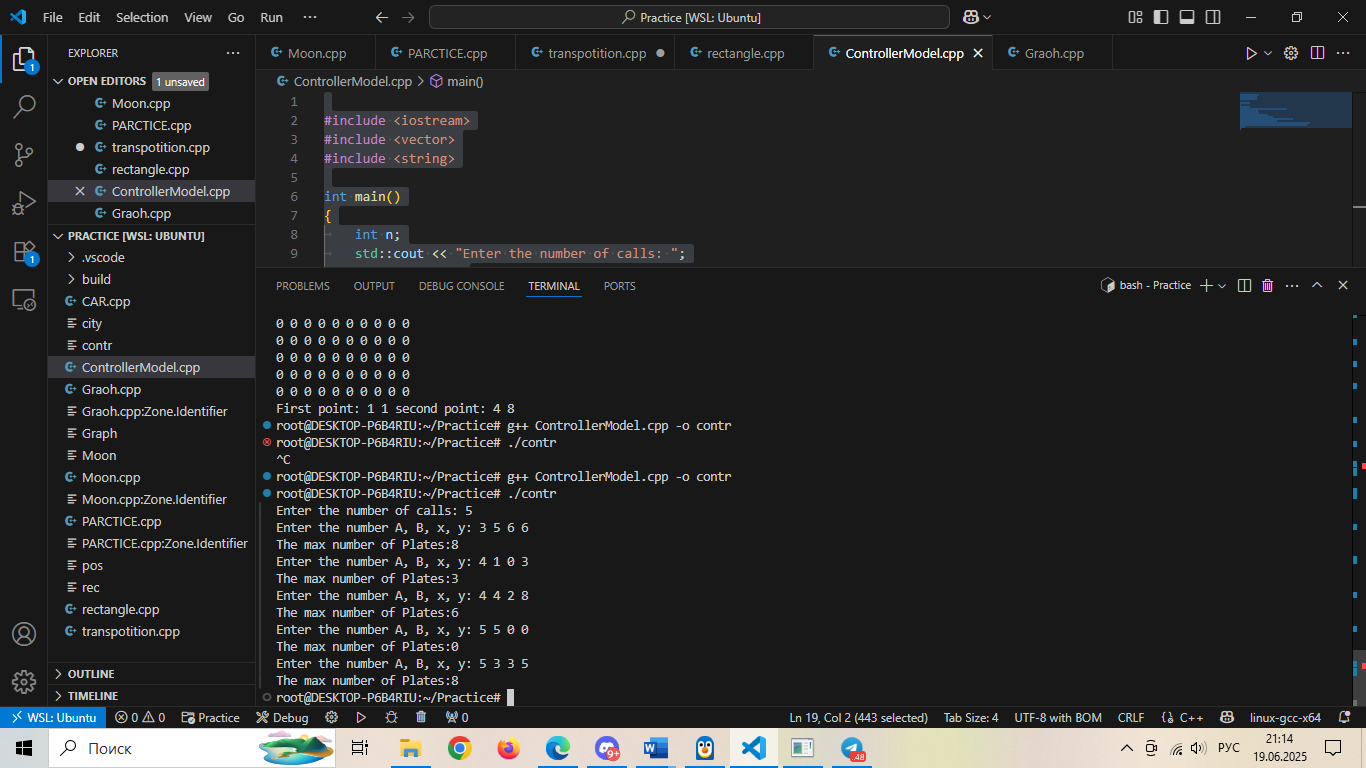


Рисунок 17 – тесты

# Задание на практику № 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.

**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.



Рисунок 18 – пример ввода-вывода

#include <iostream>

#include <vector>

#include <string>

int main()

{

    std::string word;

    std::cout << "Enter number: ";

    std::cin >> word;

    std::cout << "Is this a car number?\n";

    std::cout << "The answer is ";

    if (word.size() != 6) {

        std::cout << "NO\n";

        return 0;

    }

    for (int i = 0; i < 6; i++) {

        if ((i >= 1 && i <= 3) && !(static\_cast<int>(word[i]) <= 57 && static\_cast<int>(word[i]) >= 48)){

            //std::cout << " " << word[i] << static\_cast<int>(word[i]);

            std::cout << "NO\n";

            return 0;

        }

        if ((i == 0 || i == 4 || i == 5) && !(static\_cast<int>(word[i]) <= 90 && static\_cast<int>(word[i]) >= 65)) {

            //std::cout << " " << word[i] << static\_cast<int>(word[i]);

            std::cout << "NO\n";

            return 0;

        }

    }

    std::cout << "YES\n";

    return 0;

}

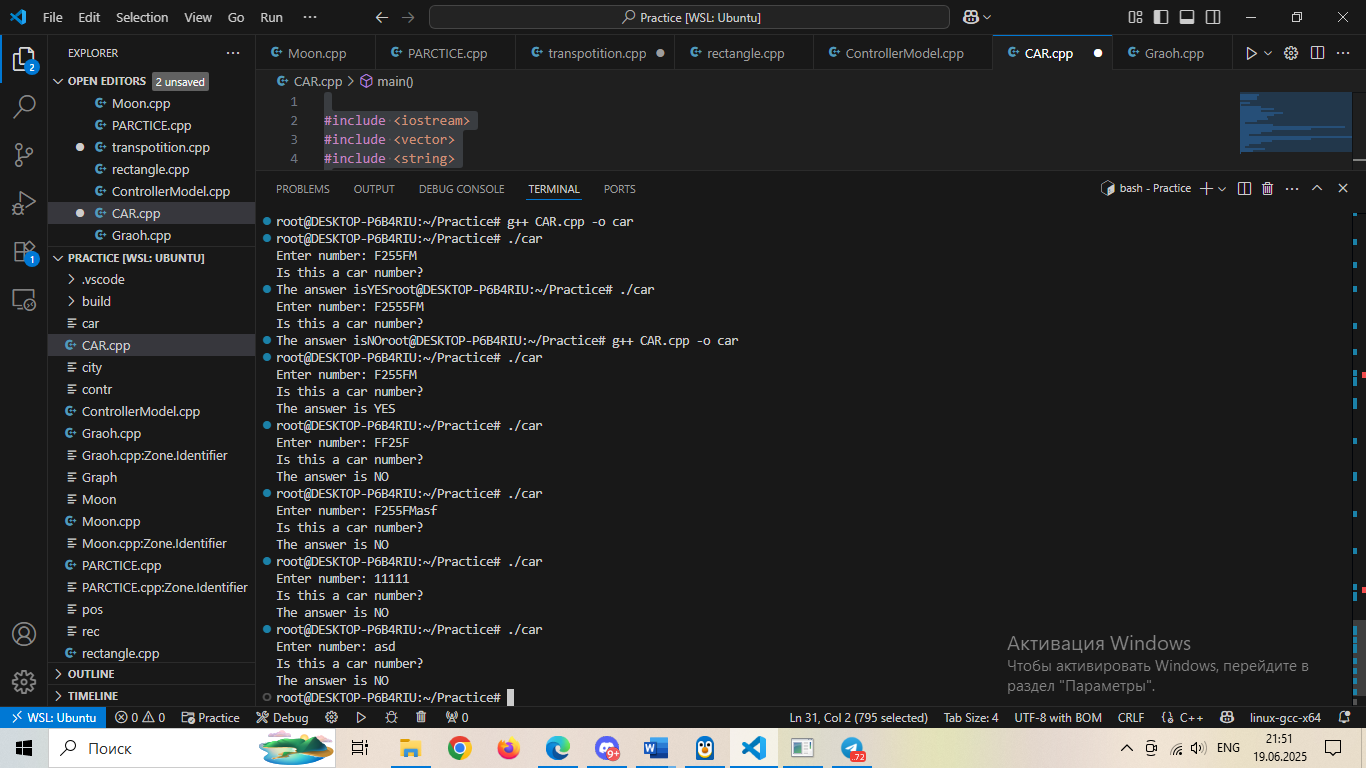


Рисунок 19 – тесты

# Задание на практику № 8

Составить светодиодную матрицу размером не менее 8 на 8 светодиодов (пример на рисунке ниже размером 4 на 4)

!Обратите внимание на ориентацию светодиодов на поле



Рисунок 20 – пример схемы

На матрицу вывести инфографику с различными динамично меняющимися изображениями.

#include "Adafruit\_NeoPixel.h"

#define PIN  3

#define NUMPIXELS 64

Adafruit\_NeoPixel strip (NUMPIXELS, PIN, NEO\_GRB + NEO\_KHZ800);

const uint32\_t letterZ[8][8] = {

  {1, 1, 1, 1, 1, 1, 1, 1},

  {1, 1, 1, 1, 1, 1, 1, 1},

  {0, 0, 1, 1, 0, 0, 0, 0},

  {0, 0, 0, 1, 1, 0, 0, 0},

  {0, 0, 0, 0, 1, 1, 0, 0},

  {0, 1, 1, 0, 0, 0, 0, 0},

  {1, 1, 1, 1, 1, 1, 1, 1},

  {1, 1, 1, 1, 1, 1, 1, 1}

};

void DRAWZ() {

  for (int row = 0; row < 8; row++) {

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

      if (letterZ[row][col]) {

        if (row == 0 || row == 1) {

          strip.setPixelColor(row \* 8 + col, strip.Color(255, 255, 255));

        } else if (row == 7 || row == 6) {

          strip.setPixelColor(row \* 8 + col, strip.Color(0, 0, 255));

        } else {

          strip.setPixelColor(row \* 8 + col, strip.Color(255, 0, 0));

        }

      }

    }

    strip.show();

    delay(100);

  }

}

void setup() {

   strip.begin();

   strip.setBrightness(50);

}

void loop() {

  for (int i = 0; i <= NUMPIXELS / 2; i++) {

    strip.setPixelColor(i, strip.Color(255, 255, 255));

    strip.setPixelColor(NUMPIXELS - i, strip.Color(255, 0, 0));

    strip.show();

    delay(100);

  }

  for (int i = 0; i < 8; i++) {

    strip.setPixelColor(24 + i, strip.Color(0, 0, 255));

    strip.setPixelColor(32 + i, strip.Color(0, 0, 255));

    strip.show();

    delay(100);

  }

  delay(200);

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

    strip.setPixelColor(i, strip.Color(0, 0, 0));

  }

  DRAWZ();

  delay(200);}

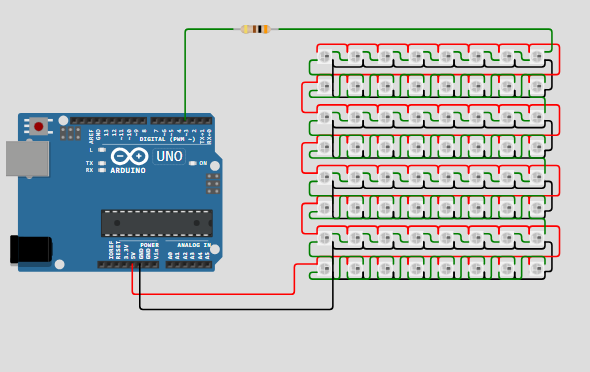


Рисунок 21 – схема

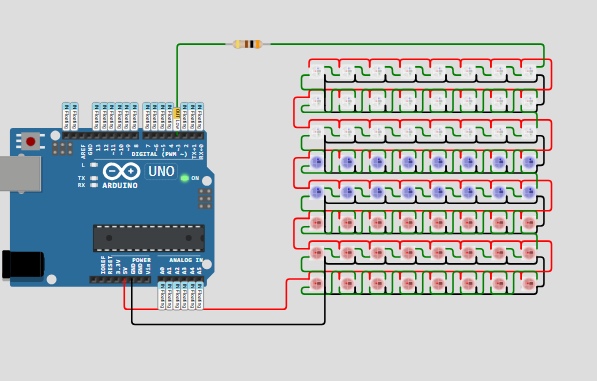


Рисунок 22 – пример работы

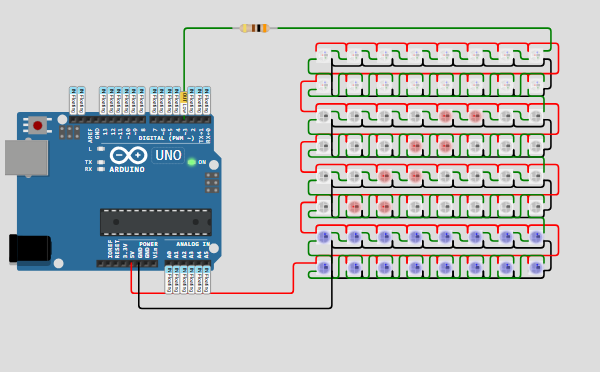


Рисунок 22 – пример работы

# Задание на практику № 9

**Задачи:**

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

Схема приведена на рисунке 1.

Зеленый светодиод – двери отрываются.

Красный светодиод ­– двери закрываются.

Фоторезистор имитируют процесс приближения-удаления человека от дверей.



Рисунок 23 – Схема управления работой автоматических дверей

Изменение значений фоторезистора осуществляется при помощи ползунка (рисунок 24), изменение значения фоторезистора доступно только, когда запущен процесс моделирования.



Рисунок 24 – Изменение значения фоторезистора

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

#define RED 9

#define GREEN 5

#define P A0

#include <LiquidCrystal\_I2C.h>

LiquidCrystal\_I2C lcd(0x27, 20, 4);

void setup() {

**Serial**.begin(115200);

pinMode(RED, OUTPUT);

pinMode(GREEN, OUTPUT);

pinMode(P, OUTPUT);

digitalWrite(RED, HIGH);

}

void loop() {

  if (analogRead(P) > 512) {

**Serial**.println("Opening door");

    digitalWrite(RED, LOW);

    digitalWrite(GREEN, HIGH);

    delay(500);

    while (analogRead(P) > 512) {

      delay(500);

    }

**Serial**.println("Closing door");

    digitalWrite(RED, HIGH);

    digitalWrite(GREEN, LOW);

  }

}

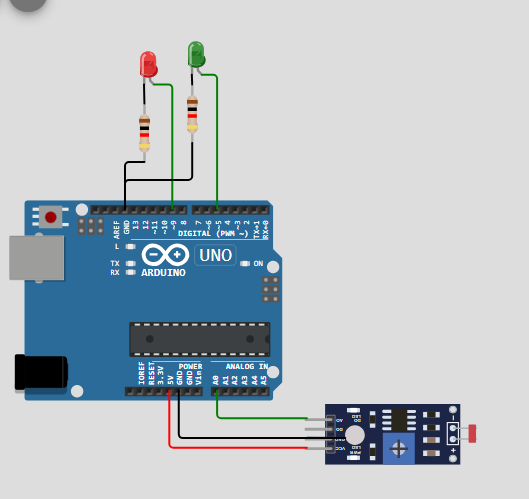


Рисунок 25 – схема

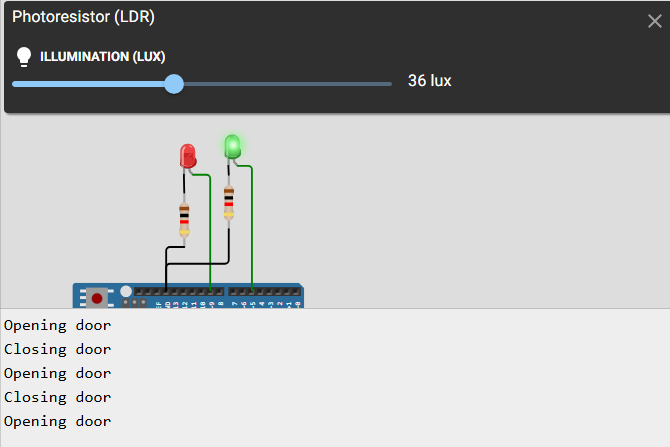


Рисунок 26 – пример открытия двери

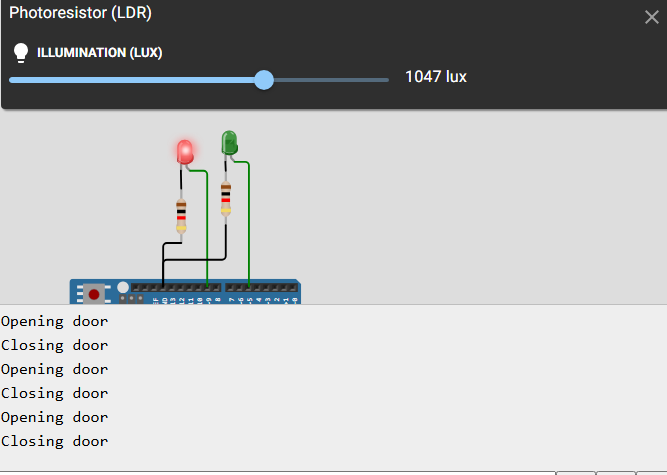


Рисунок 27 – пример закрытия двери

# Задание на практику № 10

**Задачи:**

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



Рисунок 28 – Подключение сервопривода

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.

Материалы для выполнения задания:

#include <Servo.h>

Servo myservo;

void setup() {

**Serial**.begin(115200);

  myservo.attach(3);

**Serial**.println("Enter your angle: ");

}

void loop() {

  int angle, never;

  if (**Serial**.available() > 0) {

    angle = **Serial**.parseInt();

    if (angle >= 0 && angle <= 180) {

      if (angle > myservo.read()){

**Serial**.println("going right");

          for (int i = myservo.read(); i <= angle; i++) {

            myservo.write(i);

            delay(10);

          }

      } else if (angle < myservo.read()){

**Serial**.println("going left");

        for (int i = myservo.read(); i >= angle; i--) {

          myservo.write(i);

          delay(10);

        }

      } else {

**Serial**.println("You already on this angle");

      }

    } else {

**Serial**.println("wrong angle");

    }

    while (**Serial**.available() > 0) **Serial**.read();

  }

}

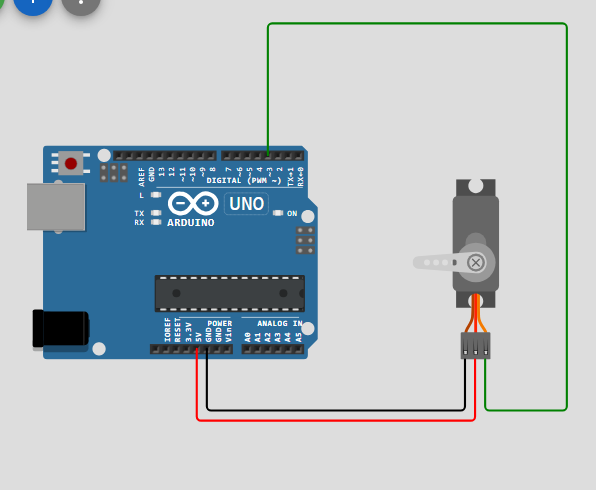


Рисунок 29 – схема

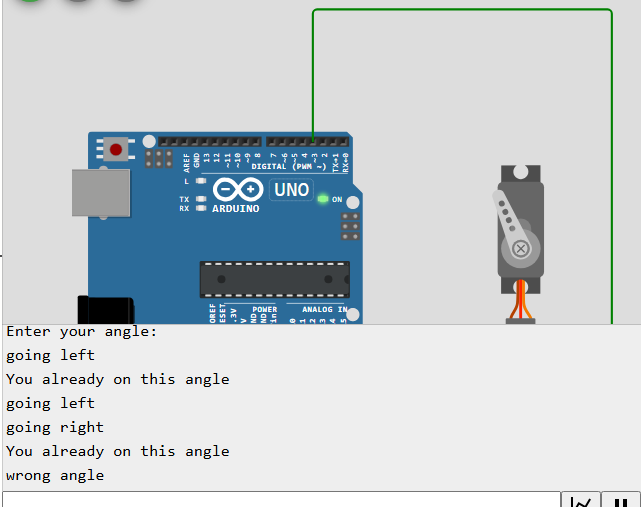


Рисунок 30 – Пример работы

# Задание на практику № 11

**Задача:**

Найдите все зеленые объекты на изображении. Найдите центры зеленых объектов. Отметьте центр красной точкой.

import cv2  
import numpy as np  
image = cv2.imread("image.jpg")  
cv2.imshow("original", image)  
cv2.waitKey(0)  
blurred\_image = cv2.GaussianBlur(image, (11, 11), 0)  
cv2.imshow("blurred", blurred\_image)  
cv2.waitKey(0)  
hsv\_image = cv2.cvtColor(blurred\_image, cv2.COLOR\_BGR2HSV)  
cv2.imshow("hsv", hsv\_image)  
cv2.waitKey(0)  
hsv\_min = np.array((36, 25, 25), np.uint8)  
hsv\_max = np.array((70, 255, 255), np.uint8)  
green\_mask = cv2.inRange(hsv\_image, hsv\_min, hsv\_max)  
contours, hierarchy = cv2.findContours(green\_mask.copy(), cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  
sorted\_contur = sorted(contours, key=cv2.contourArea, reverse=True)  
for i in sorted\_contur:  
 if cv2.contourArea(i) < 150:  
 continue  
 x, y, w, h, = cv2.boundingRect(i)  
 cv2.circle(image, (x + w // 2, y + h // 2), 10, (0, 0, 255), 3)  
cv2.imshow('contours', image)  
cv2.waitKey(0)  
cv2.destroyAllWindows()

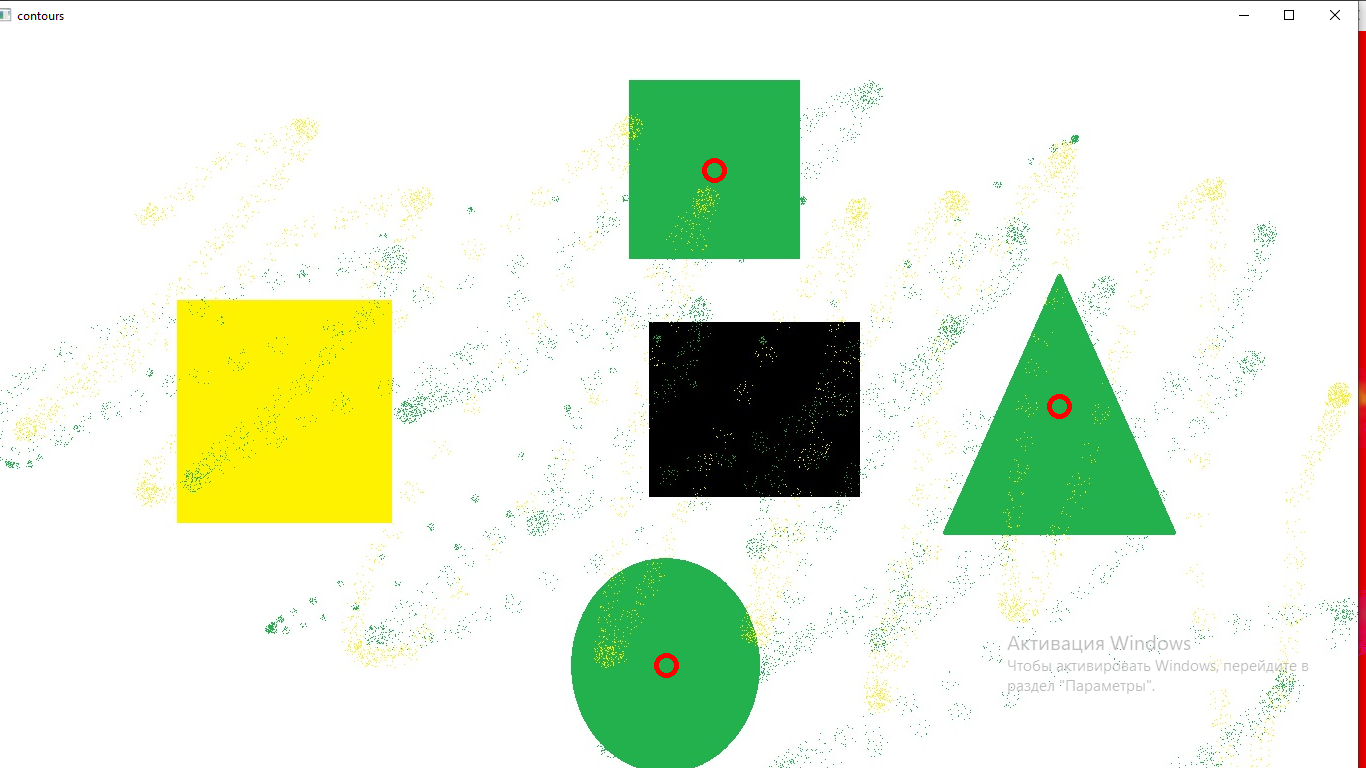


Рисунок 31 – Пример работы

# Отчет по экскурсии по АО «СО ЕЭС» 22.06.2025

22 июня наша группа отправилась на экскурсию в Филиал АО «СО ЕЭС» «Региональное диспетчерское управление энергосистемы Иркутской области». На этой экскурсии нам рассказали работу данной организации и её историю, так же нам показали некоторые кабинеты: тренировочный зал с маленьким экраном, серверную, и огромную диспетчерскую комнату с большим экраном.

Системный оператор должен выдавать оперативные диспетчерские команды и распоряжения, обязательные для исполнения субъектами электроэнергетики и потребителями электрической энергии, которые влияют на электроэнергетический режим работы энергетической системы.

Экскурсия впечатлила меня масштабом компании и количеством технологий, которыми она пользуется, но я бы не хотел работать в этой компании.

Данная экскурсия дала ценный опыт о том где могут пригодится IT специалисты после выпуска из вуза, но я бы не хотел работать в данной отрасли.

# Отчет по экскурсии в АО «ИСПсистем» 24.06.2025

24 июня наша группа сходила на экскурсию в IT компанию ИТ систем, которая разрабатывает софты. Экскурсия началась с презентации и истории компании и чем она занимается.

Эта компания занимается разработкой платформ для комплексного управления ИТ-инфраструктурой. Мы познакомились какие продукты она разрабатывает и поддерживает. У компании более 200 000 серверов в более чем в 100 странах.

Экскурсия мне понравилась, я увидел, что даже в родном городе можно найти хорошую работу в сфере IT и хотел бы пройти стажировку в этой компании.

# Заключение

За время учебной практики были успешно закреплены теоретические знания и приобретены практические навыки программирования. Основное внимание уделялось решению алгоритмических задач на C++ и Python, работе с микроконтроллерами Arduino, а также освоению принципов компьютерного зрения.

Экскурсии на предприятия АО «СО ЕЭС» и АО «ИСПсистем» позволили познакомиться с реальными задачами IT-специалистов и определиться с профессиональными предпочтениями.

Практика показала важность не только написания кода, но и тестирования, анализа эффективности и документации. Все поставленные задачи выполнены, полученный опыт станет основой для дальнейшего профессионального роста.

# Список литературы

1. Wokwi Arduino Simulator: онлайн-симулятор Arduino-проектов [Электронный ресурс]. – URL: https://wokwi.com/arduino
2. AlexGyver: подробное руководство по работе со светодиодными лентами WS2812B [Электронный ресурс]. – URL: https://alexgyver.ru/ws2812\_guide
3. AlexGyver: руководство по работе со светодиодными матрицами [Электронный ресурс]. – URL: https://alexgyver.ru/matrix\_guide
4. AlexGyver: работа с аналоговыми входами Arduino [Электронный ресурс]. – URL: https://alexgyver.ru/lessons/analog-pins
5. AlexGyver: работа с последовательным портом Serial [Электронный ресурс]. – URL: https://alexgyver.ru/lessons/serial
6. AlexGyver: подключение и управление сервоприводами [Электронный ресурс]. – URL: <https://alexgyver.ru/lessons/servo>