

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

## ОТЧЁТ

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)



Qr - код на сайт  
superjob.ru с  
резюме



Qr - код на сайт  
hh.ru с резюме

Обучающегося Симонов А.А., ИСИБ-24-1 *Am*  
(ФИО, группа, подпись)

Руководитель практики от института ИТиАД  
Кононенко Р.В., доцент института ИТиАД *Tho*  
(ФИО, должность, подпись)

Руководитель образовательной программы  
Кононенко Р.В., доцент института ИТиАД *Tho*  
(ФИО, должность, подпись)

Оценка по практике зачтено  
Кононенко Р.В. Tho 30.09.2025  
(ФИО, подпись, дата)

Содержание отчета на 41 стр.  
Приложение к отчету на 0 стр.

## Содержание

Содержание .....	1
Введение .....	2
Задание №1 .....	3
Задание №2 .....	7
Задание №3 .....	10
Задание № 4 .....	13
Задание № 5 .....	16
Задание № 6 .....	18
Задание № 7 .....	20
Задание № 8 .....	22
Задание № 9 .....	26
Задание № 10 .....	31
Задание № 11 .....	35
Итоги по экскурсии в компанию ISPsystem .....	37
Итоги по экскурсии в компанию АО «СО ЕЭС» Иркутское РДУ .....	38
Заключение .....	39
Список литературы .....	40

## **Введение.**

Учебная практика представляет собой значимый этап в образовательном процессе, который предоставляет студентам возможность применить свои теоретические знания в реальных условиях. Этот процесс способствует более глубокому пониманию особенностей будущей профессии и помогает развить навыки, необходимые для успешной профессиональной деятельности. Практика дает возможность студентам соприкоснуться с реальным рабочим окружением, что значительно повышает их уровень подготовки.

Проходя практику, учащиеся не только закрепляют теоретические знания, но и учатся работать в команде. Они приобретают навыки взаимодействия с коллегами, обсуждения проектов и совместного принятия решений. В этом процессе развиваются важные профессиональные качества, такие как критическое мышление, умение решать проблемы и целеполагание.

Кроме того, практическая деятельность способствует личностному росту студентов. Столкновение с реальными рабочими задачами, необходимость быстрого принятия решений и умение нести ответственность способствуют формированию уверенности в собственных силах и развитию лидерских навыков. Таким образом, учебная практика не только дополняет профессиональный опыт учащихся, но и играет ключевую роль в их личностном развитии.

В данной работе я поделюсь своим опытом прохождения учебной практики, проанализирую полученные знания и навыки, а также рассмотрю сложности и достижения, с которыми столкнулся в процессе. Надеюсь, что этот опыт будет полезен не только мне, но и будущим коллегам, которые окажутся перед похожими задачами в своей профессиональной жизни.

## 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. Пример наиболее короткого маршрута выделен красным цветом, сумма чисел = 10.

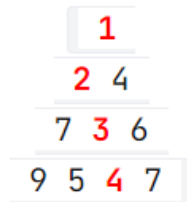


Рисунок 1 – пример горы

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 <vector>
```

```
#include <cstdlib>
```

```
#include <algorithm>
```

```
std::vector<std::vector<int>> Generate_Moon(int n) {
```

```
    std::vector<std::vector<int>> pyr(n);
```

```
    for (int i = 0; i < n; ++i) {
```

```
        pyr[i].resize(i + 1);
```

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

```
            pyr[i][j] = std::rand() % 100 + 1;
```

```
        }
```

```
    }
```

```
    return pyr;
```

```
}
```

```
std::pair<int, std::vector<int>> min_path(const std::vector<std::vector<int>>& Moon)
```

```
{
```

```

int height = Moon.size();
std::vector<std::vector<int>> Dp_Moon(height);
for (int i = 0; i < height; i++) {
    Dp_Moon[i].resize(i + 3, 99999999);
}
Dp_Moon[0][0] = Moon[0][0];
for (int i = 1; i < height; ++i) {
    for (int 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;
    }
}
int current = std::min_element(Dp_Moon[height-1].begin(), Dp_Moon[height-1].end()) - Dp_Moon[height-1].begin();
int time = Dp_Moon[height - 1][current];

std::vector<int> Path;
Path.push_back(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());
return {time, Path};
}

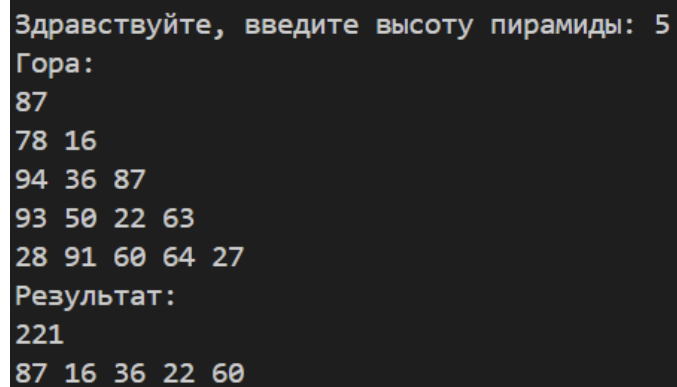
int main() {
    std::cout << "Здравствуйтесь, Введите высоту пирамиды: ";
    int n;
    std::cin >> n;
    std::rand();
    auto pyr = Generate_Moon(n);
    std::cout << "Гора:" << std::endl;
}

```

```

for (const auto& row : pyr) {
    for (size_t i = 0; i < row.size(); ++i) {
        std::cout << row[i] << (i < row.size() - 1 ? " " : "");
    }
    std::cout << "\n";
}
std::cout << "Результат:" << std::endl;
auto result = min_path(pyr);
std::cout << result.first << "\n";
for (size_t i = 0; i < result.second.size(); ++i) {
    std::cout << result.second[i] << (i < result.second.size() - 1 ? " " : "");
}
std::cout << "\n";
return 0;
}
Результат:

```



```

Здравствуйте, введите высоту пирамиды: 5
Гора:
87
78 16
94 36 87
93 50 22 63
28 91 60 64 27
Результат:
221
87 16 36 22 60

```

Рисунок 2 – пример работы

```

Здравствуйте, введите высоту пирамиды: 15
Гора:
87
78 16
94 36 87
93 50 22 63
28 91 60 64 27
41 27 73 37 12 69
68 30 83 31 63 24 68
36 30 3 23 59 70 68 94
57 12 43 30 74 22 20 85 38
99 25 16 71 14 27 92 81 57 74
63 71 97 82 6 26 85 28 37 6 47
30 14 58 25 96 83 46 15 68 35 65 44
51 88 9 77 79 89 85 4 52 55 100 33 61
77 69 40 13 27 87 95 40 96 71 35 79 68 2
98 3 18 93 53 57 2 81 87 42 66 90 45 20 41
Результат:
553
87 16 36 22 60 37 31 23 30 14 26 46 4 40 81

```

Рисунок 3 – пример работы

```

Здравствуйте, введите высоту пирамиды: 3
Гора:
87
78 16
94 36 87
Результат:
139
87 16 36

```

Рисунок 3 – пример работы

## 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.

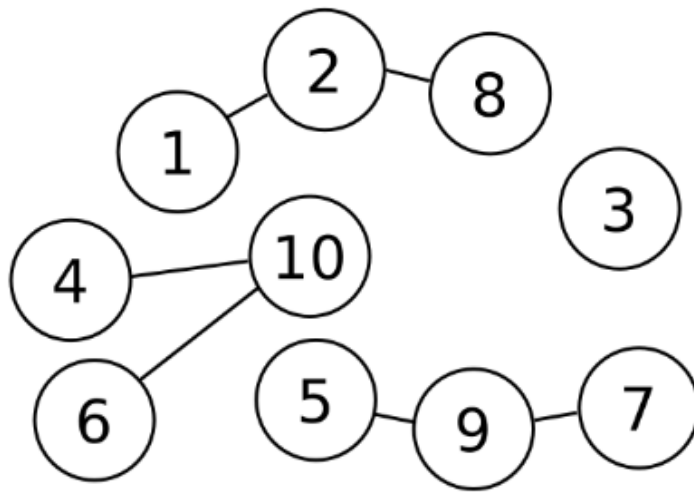


Рисунок 5 – компьютерная сеть с 3 частями

### 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>
using namespace std;
vector<vector<int>> generateGraph(int n, int m) {
    vector<vector<int>> graph(n + 1);
    cout << "Введите " << m << " рёбер (формат: u v, где u и v <= "<< n << "):" << endl;
    for (int i = 0; i < m; ++i) {
        int u, v;
        cin >> u >> v;
```



```

        graph[u].push_back(v);
        graph[v].push_back(u);
    }
    return graph;
}

void dfs(int v, vector<bool>& visited, const vector<vector<int>>& graph) {
    visited[v] = true;
    for (int neighbor : graph[v]) {
        if (!visited[neighbor]) {
            dfs(neighbor, visited, graph);
        }
    }
}

int count_communication_lines(int n, const vector<vector<int>>& graph) {
    vector<bool> visited(n+1, false);
    int count = -1;
    for (int i = 1; i < n+1; ++i) {
        if (!visited[i]) {
            dfs(i, visited, graph);
            ++count;
        }
    }
    return count;
}

int main() {
    int n, m;
    cout << "Здравствуйте, введите число вершин графа (узлы сети, не более
1000)" << endl <<
    "и набор рёбер (сохранившиеся линии связи, не более 1000) через пробел и
нажмите enter:";
    cin >> n;
    cin >> m;
    vector<vector<int>> graph = generateGraph(n,m);
    if (n <= 0 or m <= 0){
        cout << "неверные данные. Пожалуйста, введите количество вершин,
превышающее 0!" << endl;
        return 0;
    }
    int count = count_communication_lines(n, graph);
    cout << "минимальное число линий связи: " << count << endl;
    return 0;
}

```

Результат:

```
Здравствуйте, введите число вершин графа (узлы сети, не более 1000)
и набор рёбер (сохранившиеся линии связи, не более 1000) через пробел и нажмите enter:10 6
Введите 6 рёбер (формат: u v, где u и v <= 10):
1 2
2 8
4 10
5 9
6 10
7 9
минимальное число линий связи: 3
```

Рисунок 6 – пример работы

```
Здравствуйте, введите число вершин графа (узлы сети, не более 1000)
и набор рёбер (сохранившиеся линии связи, не более 1000) через пробел и нажмите enter:4 3
Введите 3 рёбер (формат: u v, где u и v <= 4):
1 2
2 3
3 4
минимальное число линий связи: 0
```

Рисунок 7 – пример работы

```
Здравствуйте, введите число вершин графа (узлы сети, не более 1000)
и набор рёбер (сохранившиеся линии связи, не более 1000) через пробел и нажмите enter:10 6
Введите 6 рёбер (формат: u v, где u и v <= 10):
1 2
2 3
3 4
4 5
7 8
8 9
минимальное число линий связи: 3
```

Рисунок 8 – пример работы

### Задание №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>
using namespace std;
void snow_center(int n, int k){
    vector<int> snow_level(n,0);
    int count_k = 0;
    int command, a, b;
    while (count_k != k)
    {
```

```

cin >> command >> a >> b;
if (command == 1){
    snow_level[a-1] = b;
} else if (command == 2){
    int snow_count = 0;
    for (int i = a-1; i <= b-1; i++){
        snow_count += snow_level[i];
    }
    cout << snow_count << endl;
} else {
    cout << "Неправильные входные данные. они должны начинаться либо с 1,
либо с 2." << endl;
    continue;
}
++count_k;
}
}
int main() {
    int n,k;
    cout << "введите количество улиц и число запросов через пробел: ";
    cin >> n >> k;
    snow_center(n,k);
    return 0;
}

```

Результат:

```

введите количество улиц и число запросов через пробел: 3 8
1 3 1
1 1 3
2 1 3
1 2 7
2 2 3
1 3 9
1 3 7
2 1 3
4
8
27

```

Рисунок 9 – пример работы

```

введите количество улиц и число запросов через пробел: 6 5
2 1 6
1 3 2
2 2 4
1 6 3
2 1 6
0
2
5

```

Рисунок 10 – пример работы

```
введите количество улиц и число запросов через пробел: 5 3
1 3 7
1 1 4
2 1 5
11
```

Рисунок 11 – пример работы

## Задание № 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.

t	r	a	n	s	p	o	s	i	t	i	o	n
1	2	3	4	5	6	7	8	9	10	11	12	13
5	11	13	12	6	1	8	4	10	9	7	2	3
s	i	n	o	p	t	s	n	t	i	o	r	a
1	2	3	4	5	6	7	8	9	10	11	12	13
5	11	13	12	6	1	8	4	10	9	7	2	3
p	o	a	r	t	s	n	o	i	t	s	i	n

Рисунок 12 – Перестановка слова *transposition* 2 раза

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>
using namespace std;
```

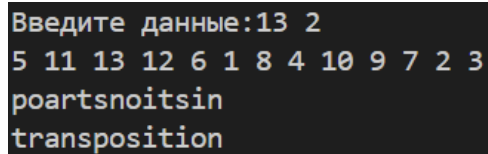
```

string decrypt(const string& encrypted, const vector<int>& permutation, int k) {
    int n = encrypted.size();
    string original = encrypted;
    vector<int> inversePermutation(n);
    for (int i = 0; i < n; ++i) {
        inversePermutation[permutation[i] - 1] = i;
    }
    for (int i = 0; i < k; ++i) {
        string temp = original;
        for (int j = 0; j < n; ++j) {
            original[j] = temp[inversePermutation[j]];
        }
    }
    return original;
}

int main() {
    int n, k;
    cout << "Введите данные:"
    cin >> n >> k;
    vector<int> permutation(n);
    for (int i = 0; i < n; ++i) {
        cin >> permutation[i];
    }
    string encrypted;
    cin >> encrypted;
    string original = decrypt(encrypted, permutation, k);
    cout << original << endl;
    return 0;
}

```

Результат:

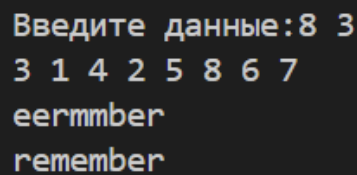


```

Введите данные:13 2
5 11 13 12 6 1 8 4 10 9 7 2 3
poartsnoitsin
transposition

```

Рисунок 13 – Пример работы



```

Введите данные:8 3
3 1 4 2 5 8 6 7
eermmber
remember

```

Рисунок 14 – Пример работы

```
Введите данные:5 3  
4 3 5 1 2  
arebd  
bread
```

Рисунок 15 – Пример работы



## Задание № 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.

### 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>
using namespace std;
int main() {
    int h, w;
    cout << "Введите высоту матрицы и её ширину через," << endl;
    cout << "а потом саму матрицу:";
    cin >> h >> w;
    vector<vector<int>> matrix(h, vector<int>(w));
    for (int i = 0; i < h; ++i) {
        for (int j = 0; j < w; ++j) {
            cin >> matrix[i][j];
        }
    }
    int minRow = h, maxRow = -1, minCol = w, maxCol = -1;

    for (int i = 0; i < h; ++i) {
        for (int j = 0; j < w; ++j) {
            if (matrix[i][j] == 1) {
                minRow = min(minRow, i);
                maxRow = max(maxRow, i);
                minCol = min(minCol, j);
                maxCol = max(maxCol, j);
            }
        }
    }
}
```

```

    }
    cout << minRow - 1 << " " << minCol - 1 << " " << maxRow + 1 << " " <<
maxCol + 1 << endl;
    return 0;
}

```

Результат:

```

Введите высоту матрицы и её ширину через,
а потом саму матрицу:5 7
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 1 1 1 1 0
0 0 1 1 1 1 0
0 0 0 0 0 0 0
1 1 4 6

```

Рисунок 16 – Пример работы

```

Введите высоту матрицы и её ширину через,
а потом саму матрицу: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 2 8 6

```

Рисунок 17 – Пример работы

```

Введите высоту матрицы и её ширину через,
а потом саму матрицу:4 4
0 0 0 0
0 1 1 0
0 0 0 0
0 0 0 0
0 0 2 3

```

Рисунок 18 – Пример работы

## Задание № 6

Постановка задачи:

В школьном кружке робототехники есть два вида микроконтроллеров (условно тип А и тип В) и два вида модулей управления мотором (условно тип 1 и тип 2). Выяснилось, что контроллер типа В и модуль управления типа 2 несовместимы. Использование микроконтроллеров и модулей управления в других комбинациях возможно. Имеется а микроконтроллеров типа А, b микроконтроллеров типа В, x модулей управления типа 1 и у 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 <vector>
#include <algorithm>
using namespace std;
int main() {
    int n;
    cout << "Введите количество запросов: ";
    cin >> n;
    vector<int> results;
    for (int i = 0; i < n; i++) {
        int a, b, x, y;
        cin >> a >> b >> x >> y;
        int pairs_A = min(a, x + y);
        int pairs_B = min(b, x);
        int total_pairs = pairs_A + pairs_B;
        results.push_back(total_pairs);
    }
    for (int i = 0; i < n; i++) {
        cout << results[i] << " ";
    }
    cout << endl;
    return 0;
}
```

Результат:

```
введите количество запросов:7
3 7 9 12
0 5 11 8
14 2 4 10
6 13 1 7
12 12 3 0
8 14 5 6
10 1 15 2
10 5 14 8 3 11 11
```

Рисунок 19 – Пример работы

```
введите количество запросов:15
3 7 9 12
0 5 11 8
14 2 4 10
6 13 1 7
12 12 3 0
8 14 5 6
10 1 15 2
4 9 3 11
7 0 15 6
5 8 2 14
1 3 10 12
6 4 13 0
15 2 7 8
9 11 5 1
2 12 6 4
10 5 14 8 3 11 11 13 7 13 4 6 15 6 6
```

Рисунок 20 – Пример работы

```
введите количество запросов:4
12 8 19 24
5 13 0 17
22 10 4 15
3 21 6 11
20 17 19 14
```

Рисунок 21 – Пример работы

## Задание № 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.

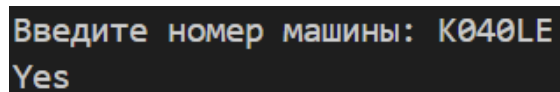
### Формат выходных данных

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

Ход решения:

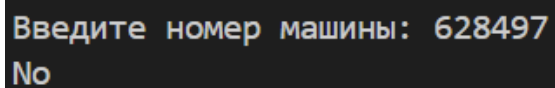
```
#include <iostream>
#include <string>
using namespace std;
bool isValidCarNumber(const string& number) {
    if (number.length() != 6) return false;
    if (!isupper(number[0])) return false;
    for (int i = 1; i <= 3; i++) {
        if (!isdigit(number[i])) return false;
    }
    for (int i = 4; i < 6; i++) {
        if (!isupper(number[i])) return false;
    }
    return true;
}
int main() {
    string carNumber;
    cout << "Введите номер машины: ";
    cin >> carNumber;
    if (isValidCarNumber(carNumber)) {
        cout << "Yes\n";
    } else {
        cout << "No\n";
    }
    return 0;
}
```

Результат:



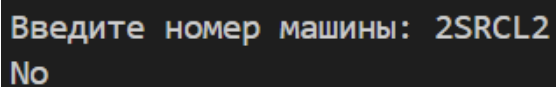
```
Введите номер машины: K040LE
Yes
```

Рисунок 22 – Пример работы



```
Введите номер машины: 628497
No
```

Рисунок 23 – Пример работы



```
Введите номер машины: 2SRCL2
No
```

Рисунок 24 – Пример работы

## Задание № 8

Постановка задачи:

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**

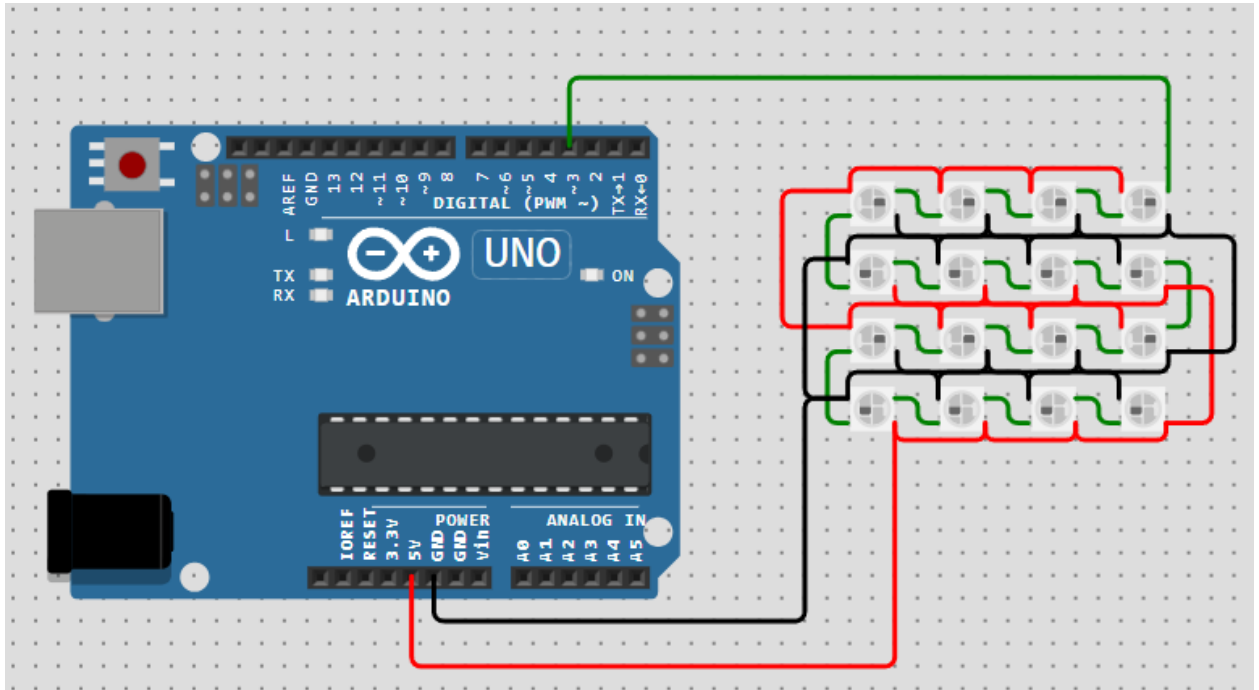


Рисунок 25 – Схема подключения матрицы 4 на 4

На матрицу вывести инфографику с различными динамично меняющимися изображениями.

Ход решения:

```
#include "Adafruit_NeoPixel.h"
```

```
#define PIN 3
```

```
#define NUMPIXELS 64
```

```
#define LETTER_DELAY 2000
```

```
#define PIXEL_DELAY 50
```

```
Adafruit_NeoPixel strip(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
```

```
const uint32_t LETTER_COLOR = strip.Color(0, 0, 255);
```

```
const uint32_t BG_COLOR = strip.Color(0, 0, 0);
```

```
const uint8_t letters[6][8][8] = {
```

```
{
```

```
{0,0,0,0,0,0,0,0},
```

```
{0,1,0,0,0,0,1,0},
```

```
{0,1,1,0,0,0,1,0},
```

```
{0,1,0,0,1,0,1,0},
```

```
{0,1,0,0,1,0,1,0},
```

```
{0,1,1,0,0,0,1,0},
```

```
}
```

```

    {0,1,0,0,0,0,1,0},
    {0,0,0,0,0,0,0,0}
  },
  {
    {0,0,0,0,0,0,0,0},
    {0,1,1,1,1,1,0,0},
    {0,1,0,0,0,0,1,0},
    {0,1,0,0,0,0,1,0},
    {0,0,1,1,1,1,1,0},
    {0,1,0,0,0,0,0,0},
    {0,0,0,0,0,0,1,0},
    {0,0,0,0,0,0,0,0}
  },
  {
    {0,0,0,0,0,0,0,0},
    {0,1,0,0,0,0,1,0},
    {0,1,0,0,0,0,1,0},
    {0,1,1,1,1,1,1,0},
    {0,1,1,1,1,1,1,0},
    {0,1,0,0,0,0,1,0},
    {0,1,0,0,0,0,1,0},
    {0,0,0,0,0,0,0,0}
  },
  {
    {0,0,0,0,0,0,0,0},
    {0,1,0,0,0,0,1,0},
    {0,1,1,0,0,0,1,0},
    {0,1,0,0,1,0,1,0},
    {0,1,0,0,1,0,1,0},
    {0,1,1,0,0,0,1,0},
    {0,1,0,0,0,0,1,0},
    {0,0,0,0,0,0,0,0}
  },
  {
    {0,0,0,0,0,0,0,0},
    {0,1,1,1,1,1,1,0},
    {0,1,1,1,1,1,1,0},
    {0,0,0,1,1,0,0,0},
    {0,0,0,1,1,0,0,0},
    {0,0,0,1,1,0,0,0},
    {0,0,0,1,1,0,0,0},
    {0,0,0,0,0,0,0,0}
  },
  {

```



```

        {0,0,0,0,0,0,0,0},
        {0,1,0,0,0,0,1,0},
        {0,1,0,0,0,0,1,0},
        {0,1,1,1,1,1,1,0},
        {0,1,0,0,0,0,0,0},
        {0,0,0,0,0,0,1,0},
        {0,1,1,1,1,1,0,0},
        {0,0,0,0,0,0,0,0}
    }
};

void clearMatrix() {
    for(int i = 0; i < NUMPIXELS; i++) {
        strip.setPixelColor(i, BG_COLOR);
    }
    strip.show();
}

void drawLetterWithPixelDelay(uint8_t letterIndex) {
    clearMatrix();
    int activePixels[64][2];
    int count = 0;
    for(int row = 0; row < 8; row++) {
        for(int col = 0; col < 8; col++) {
            if(letters[letterIndex][row][col]) {
                activePixels[count][0] = row;
                activePixels[count][1] = col;
                count++;
            }
        }
    }
    for(int i = 0; i < count; i++) {
        int row = activePixels[i][0];
        int col = activePixels[i][1];
        strip.setPixelColor(row * 8 + col, LETTER_COLOR);
        strip.show();
        delay(PIXEL_DELAY);
    }
}

void setup() {
    strip.begin();
    strip.setBrightness(100);
    clearMatrix();
    delay(1000);
}

void loop() {

```

```

for(int i = 0; i < 6; i++) {
  drawLetterWithPixelDelay(i);
  delay(LETTER_DELAY);
  clearMatrix();
  delay(200);
}
delay(1000);
}

```

Результат:

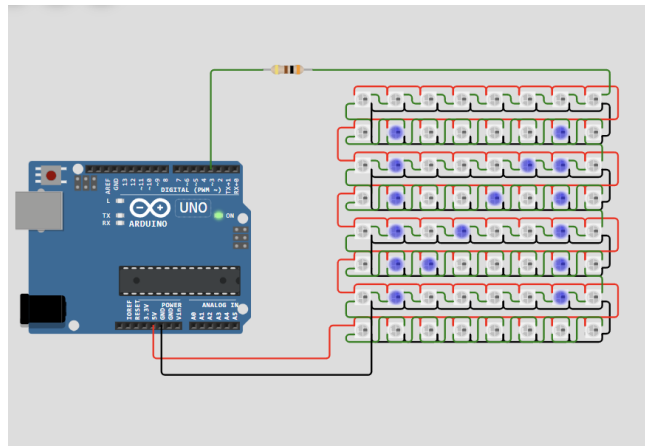


Рисунок 26 – Пример работы

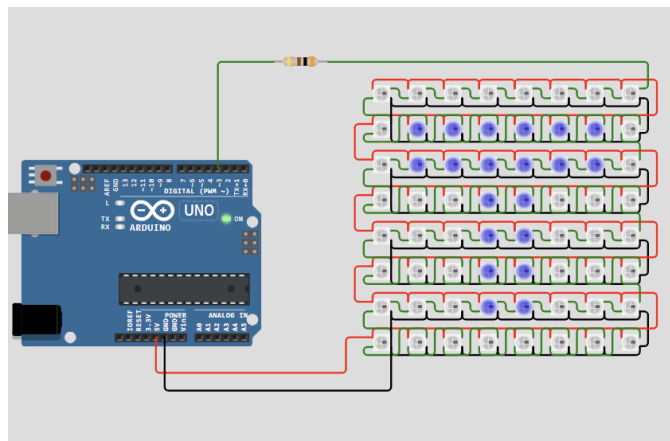


Рисунок 27 – Пример работы

Сылка на схему: <https://wokwi.com/projects/434248634094201857>

## Задание № 9

Постановка задачи:

**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 открываются.

Красный светодиод – двери закрываются.

Фоторезистор имитируют процесс приближения-удаления человека от дверей.

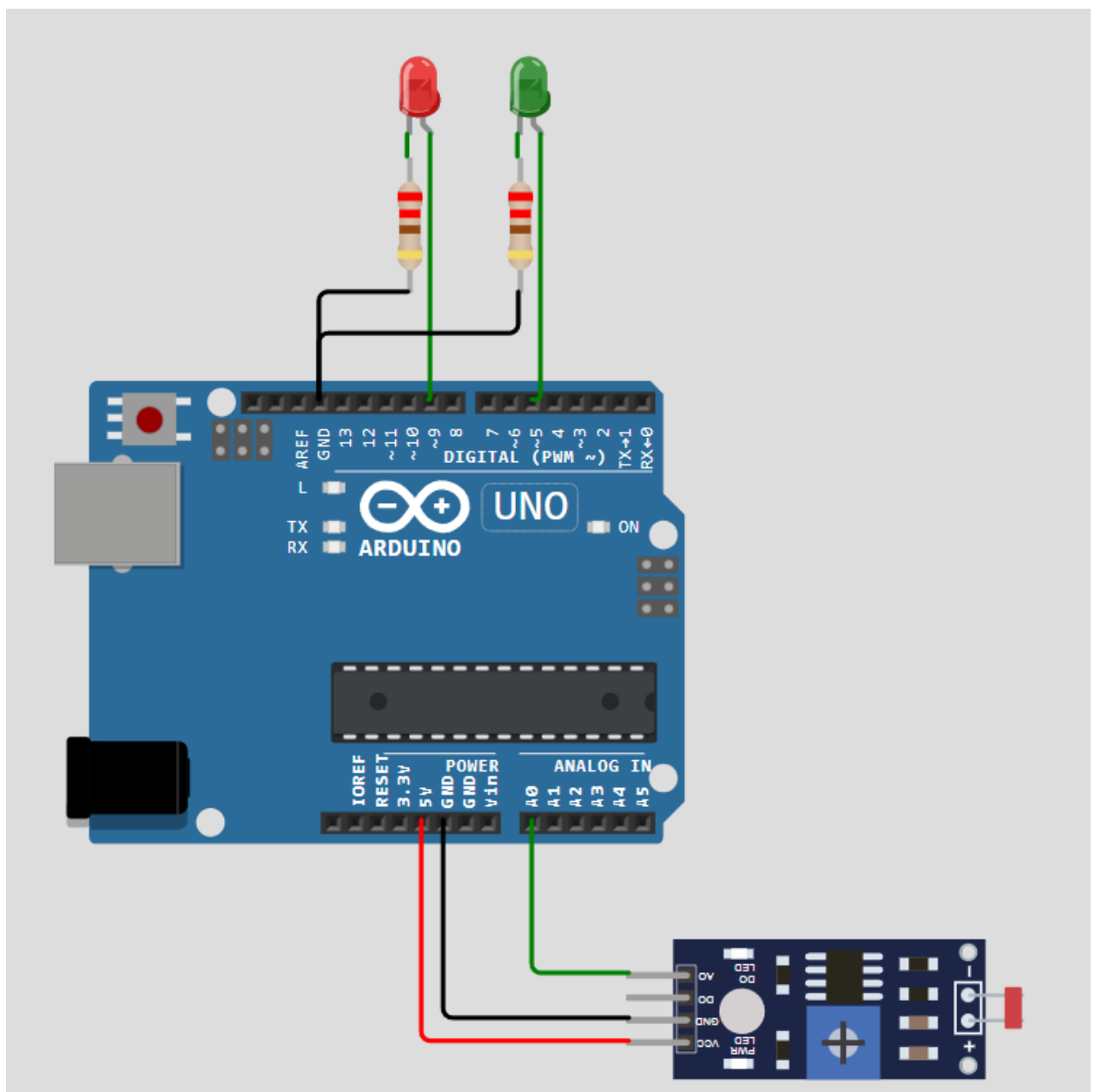


Рисунок 28 – Схема управления работой автоматических дверей

Изменение значений фоторезистора осуществляется при помощи ползунка (рисунок 2), изменение значения фоторезистора доступно только, когда запущен процесс моделирования.

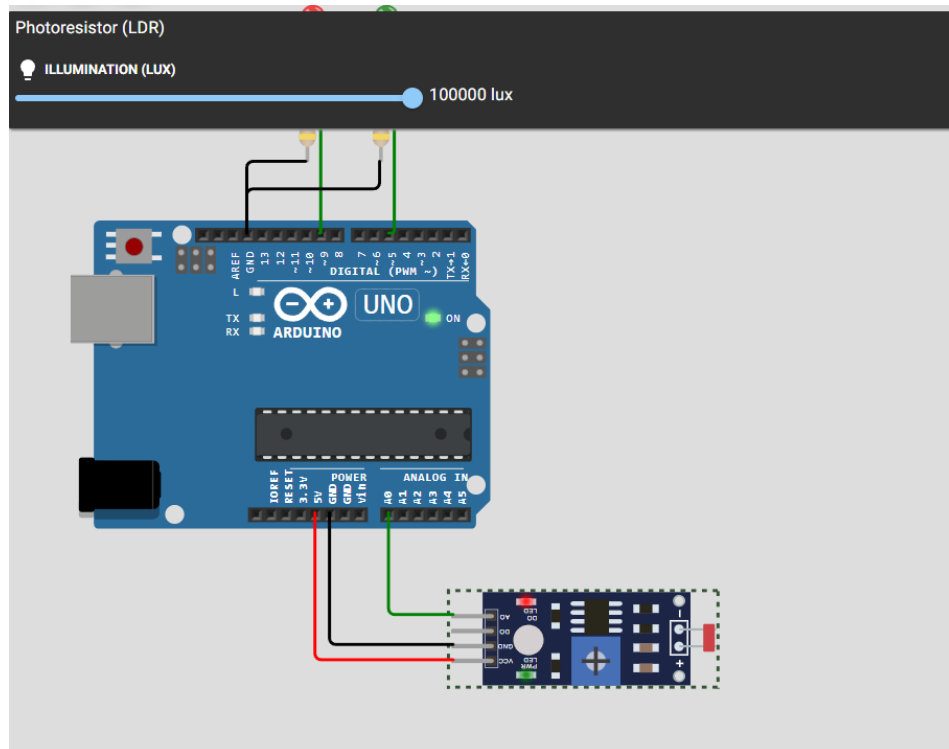


Рисунок 29 – Схема изменение значения фоторезистора

Логика работы программы:

- 1) По умолчанию горит светодиод, имитирующий закрытую дверь
- 2) Микроконтроллер считывает значение фоторезистора с аналогового 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 RED_PIN 9
#define GREEN_PIN 5
#define PHOTO_PIN A0
#define LIGHT_THRESHOLD 512
#define DOOR_OPEN_DELAY 3000
unsigned long doorOpenTime = 0;
bool doorIsOpen = false;
void setup() {
```

```

Serial.begin(115200);
pinMode(RED_PIN, OUTPUT);
pinMode(GREEN_PIN, OUTPUT);
digitalWrite(RED_PIN, HIGH);
}
void loop() {
  int lightValue = analogRead(PHOTO_PIN);
  if (lightValue > LIGHT_THRESHOLD && !doorIsOpen) {
    openDoor();
    doorOpenTime = millis();
  }
  if (doorIsOpen && (millis() - doorOpenTime >= DOOR_OPEN_DELAY)) {
    if (analogRead(PHOTO_PIN) <= LIGHT_THRESHOLD) {
      closeDoor();
    } else {
      doorOpenTime = millis();
    }
  }
}
void openDoor() {
  doorIsOpen = true;
  digitalWrite(RED_PIN, LOW);
  digitalWrite(GREEN_PIN, HIGH);
  Serial.println("Дверь открывается");
}
void closeDoor() {
  doorIsOpen = false;
  digitalWrite(GREEN_PIN, LOW);
  digitalWrite(RED_PIN, HIGH);
  Serial.println("Дверь закрывается");
}

```

Результат:

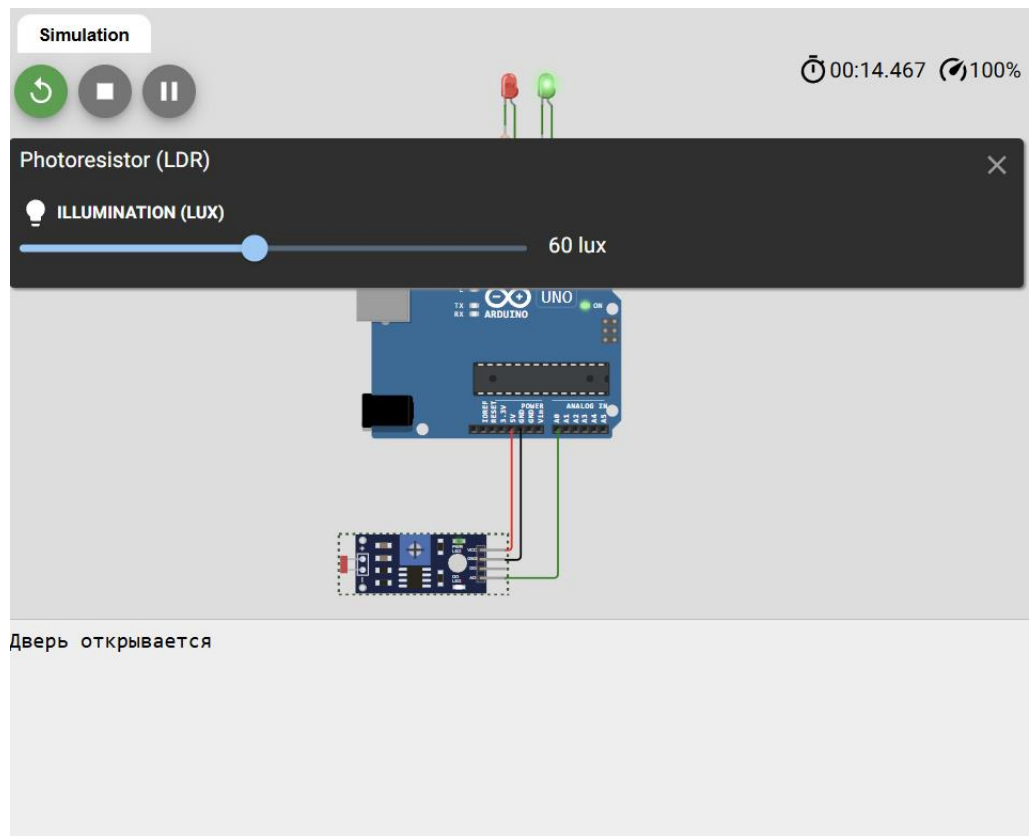


Рисунок 30 – Пример работы

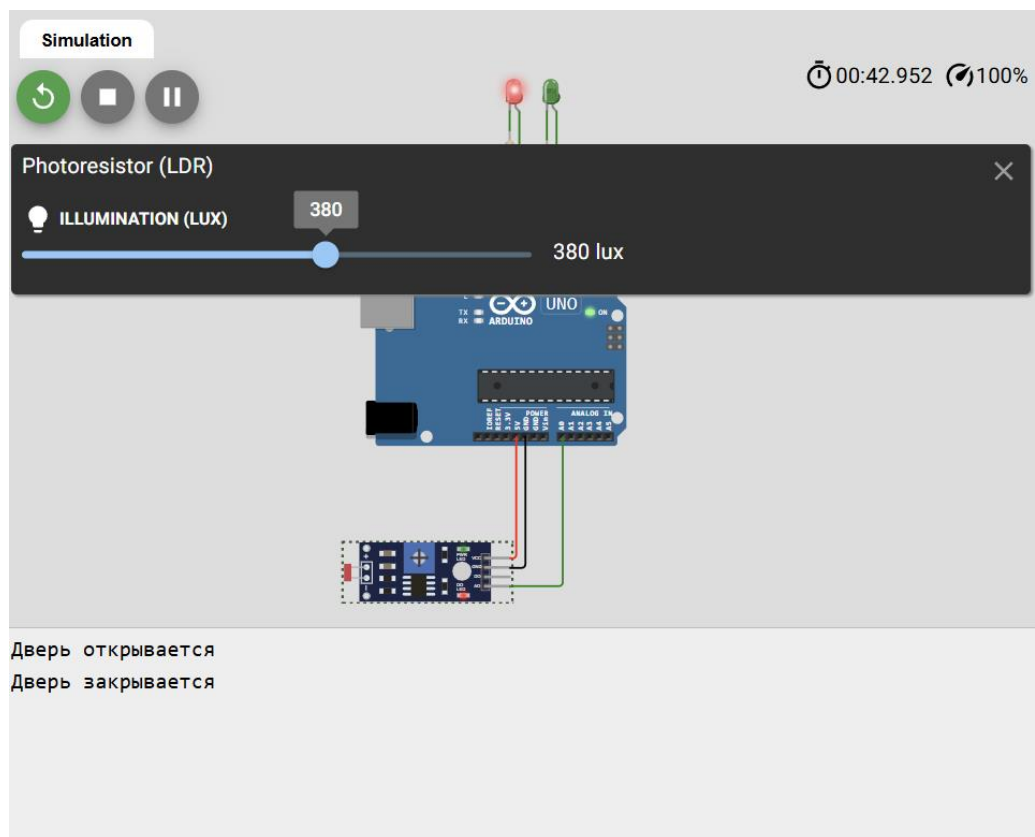


Рисунок 31 – Пример работы

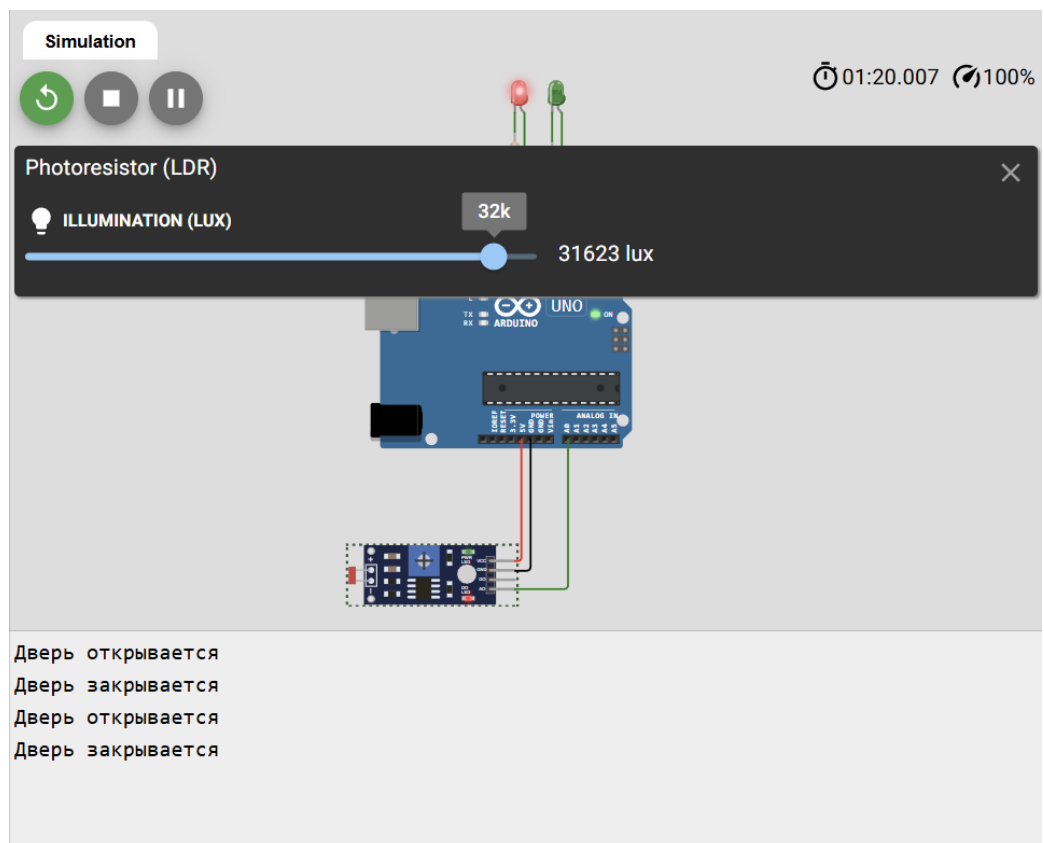


Рисунок 32 – Пример работы

Сылка на схему: <https://wokwi.com/projects/434473120345576449>

## Задание № 10

Постановка задачи:

**Задачи:**

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

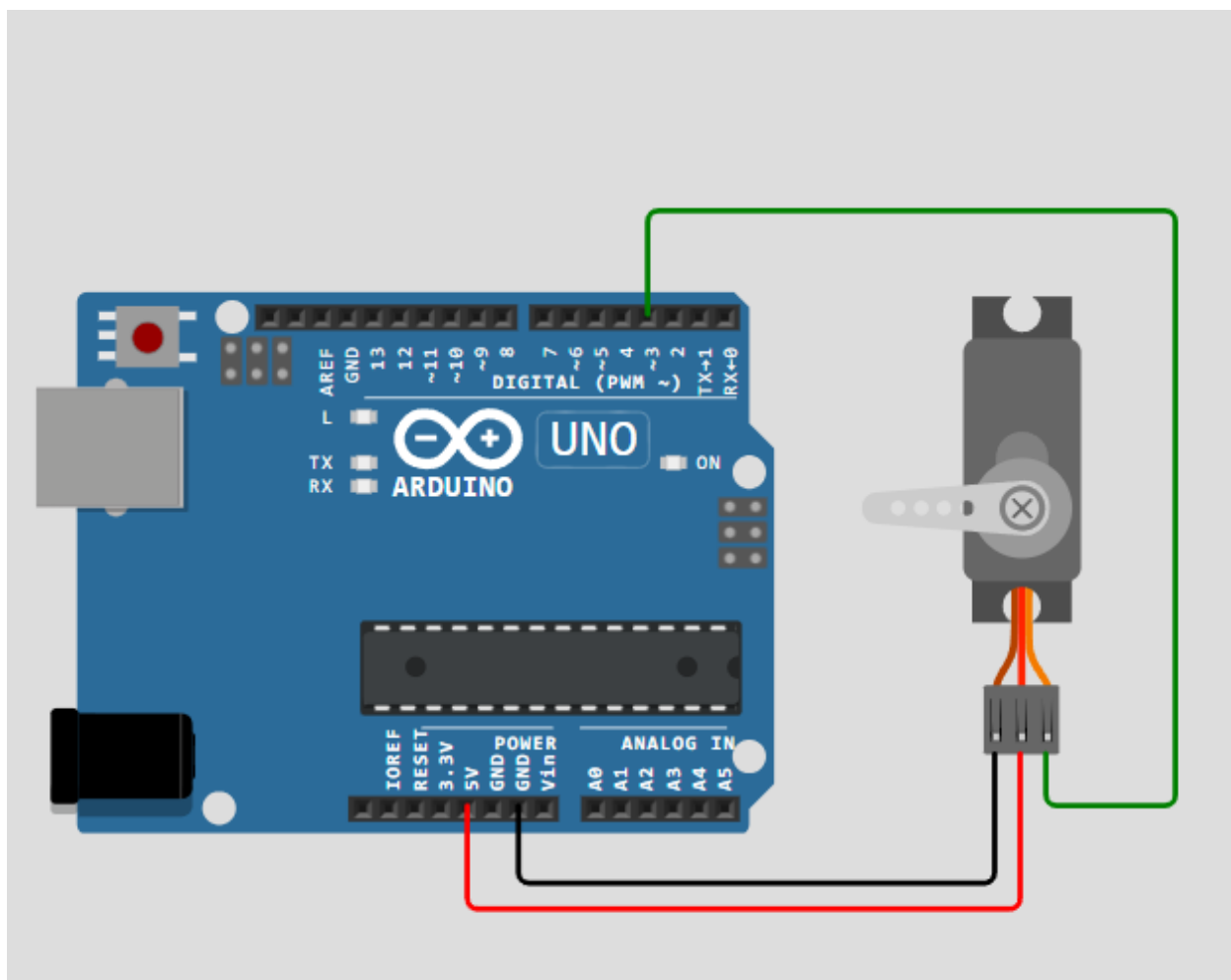


Рисунок 33 – Схема подключение сервопривода

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.

Ход решения:

```
#include <Servo.h>
```



```

Servo myservo;
int currentAngle = 90;
int targetAngle = 90;
void setup() {
  myservo.attach(3);
  Serial.begin(9600);
  Serial.println("Введите угол (0-180):");
  myservo.write(currentAngle);
}
void loop() {
  if (Serial.available() > 0) {
    String input = Serial.readStringUntil('\n');
    input.trim();

    bool isNumber = true;
    for (int i = 0; i < input.length(); i++) {
      if (!isdigit(input.charAt(i))) {
        isNumber = false;
        break;
      }
    }
    if (isNumber) {
      targetAngle = input.toInt();

      if (targetAngle >= 0 && targetAngle <= 180) {
        Serial.print("Поворот на: ");
        Serial.print(targetAngle);
        Serial.println("");

        if (targetAngle != currentAngle) {
          int step = (targetAngle > currentAngle) ? 1 : -1;
          while (currentAngle != targetAngle) {
            currentAngle += step;
            myservo.write(currentAngle);
            delay(15);
          }
        }
        else {
          Serial.println("Ошибка: угол должен быть от 0 до 180!");
        }
        else {
          Serial.println("Ошибка: введите число!");
        }
      }
    }
  }
}

```

}  
Результат:

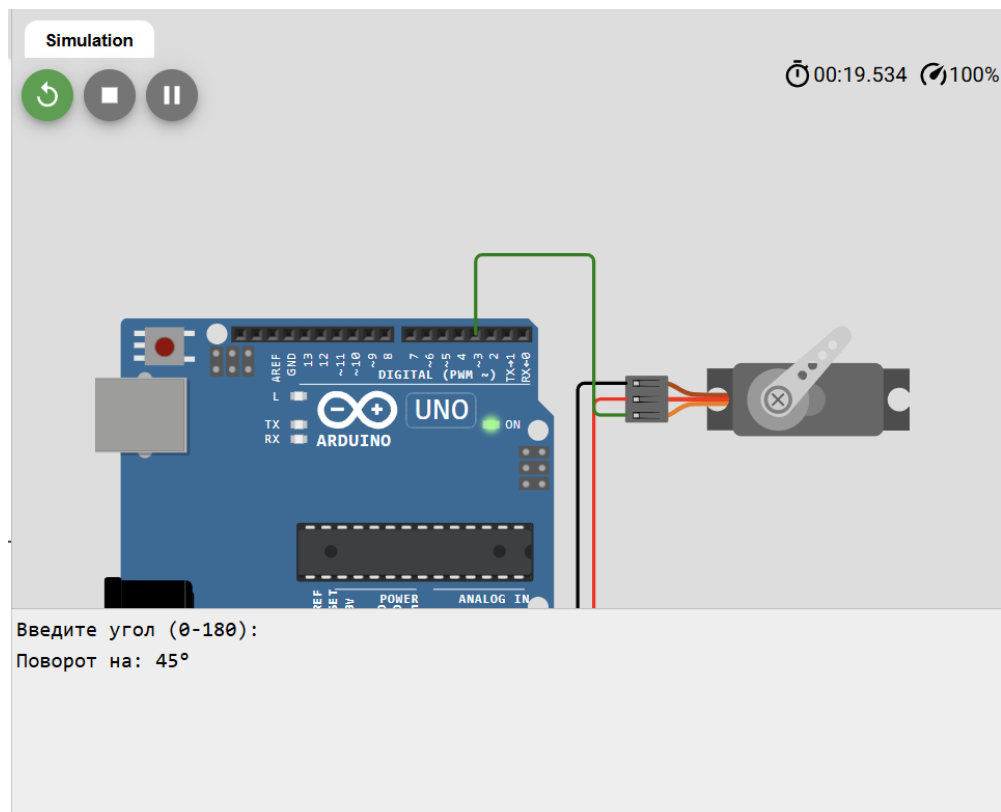


Рисунок 34 – Пример работы

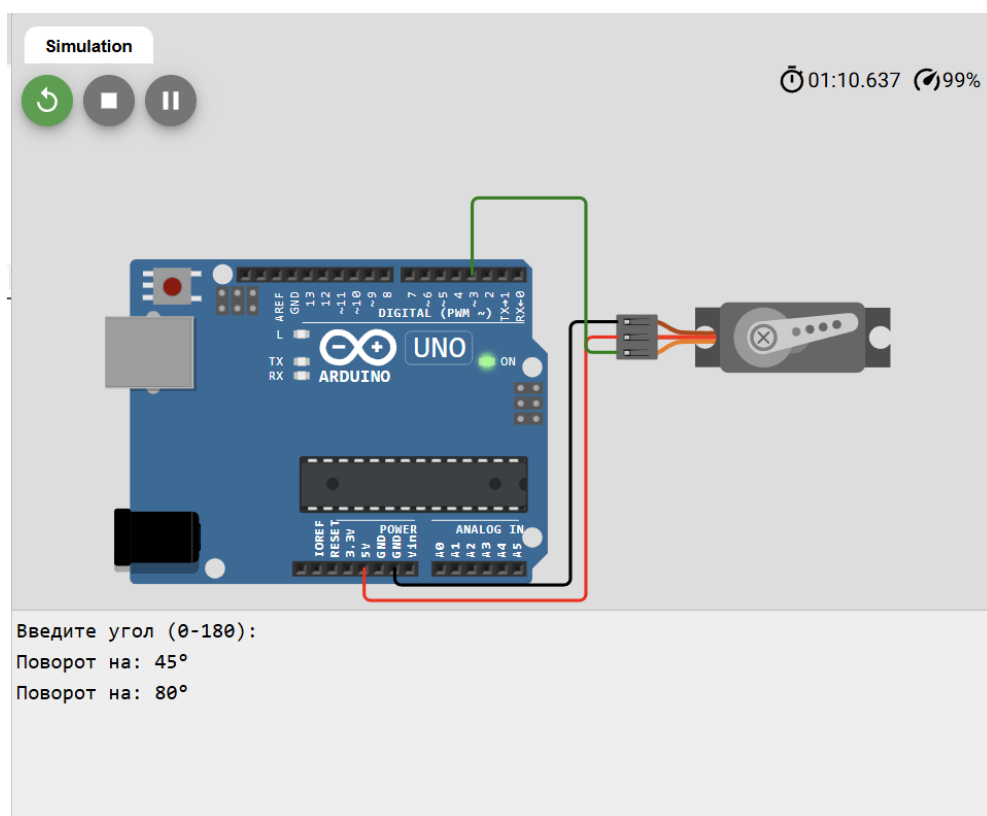


Рисунок 35 – Пример работы

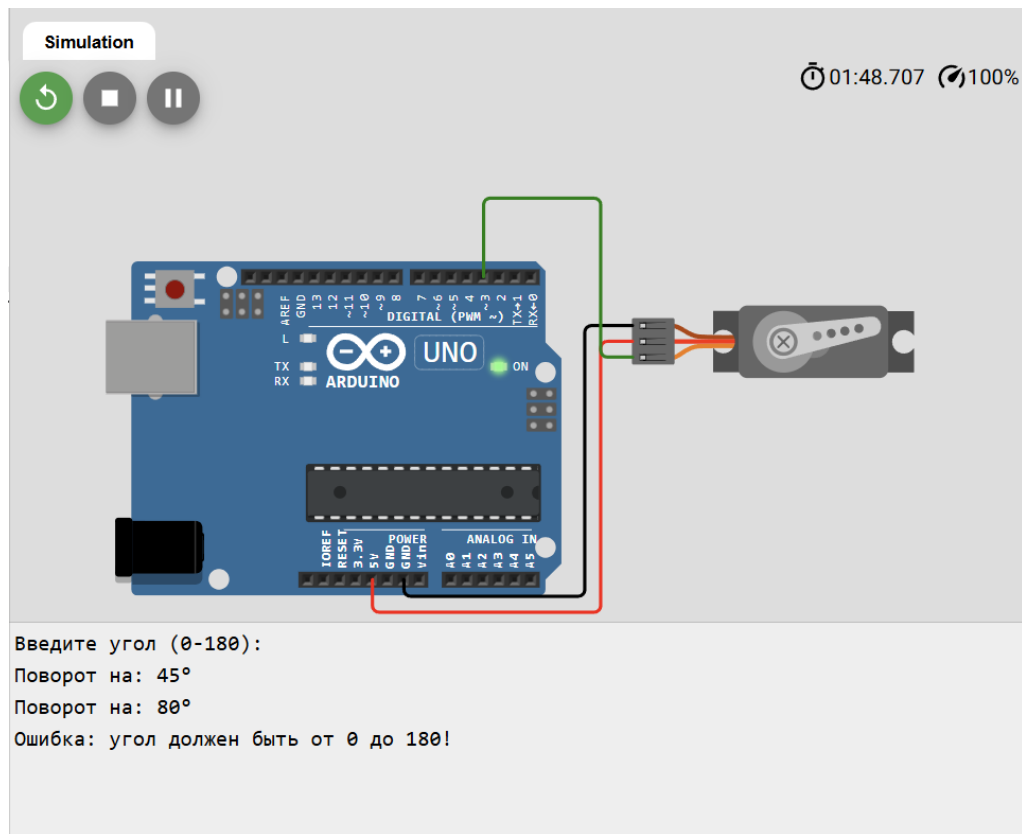


Рисунок 36 – Пример работы

Ссылка на схему: <https://wokwi.com/projects/434474188642265089>

## Задание № 11

Постановка задачи:

С помощью языка Python найдите самый крупный объект на изображении. Определите центр самого крупного объекта. Обведите самый крупный объект красной рамкой.

Ход решения:

```
import cv2
import numpy as np
image = cv2.imread("2.png")
blurred_image = cv2.GaussianBlur(image, (11, 11), 0)
hsv_image = cv2.cvtColor(blurred_image, cv2.COLOR_BGR2HSV)
hsv_min = np.array((0, 25, 25), np.uint8)
hsv_max = np.array((180, 255, 255), np.uint8)
mask = cv2.inRange(hsv_image, hsv_min, hsv_max)
contours, _ = cv2.findContours(mask.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
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), 2)
cv2.circle(image, (center_x, center_y), 5, (0, 0, 255), -1)
print("Центр прямоугольника: (", center_x, center_y, ")")
print("контур: x = ", x, "y = ", y, "width = ", w, "height=", h)
cv2.imshow('Result', image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Результат:

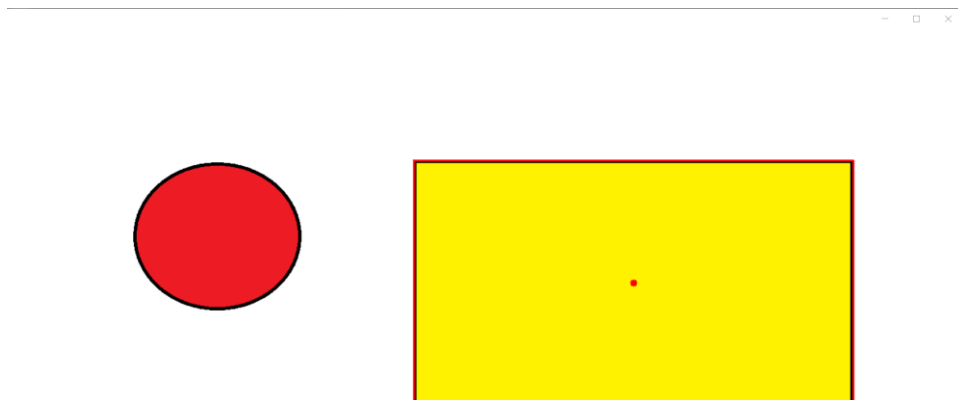


Рисунок 37 – Пример работы

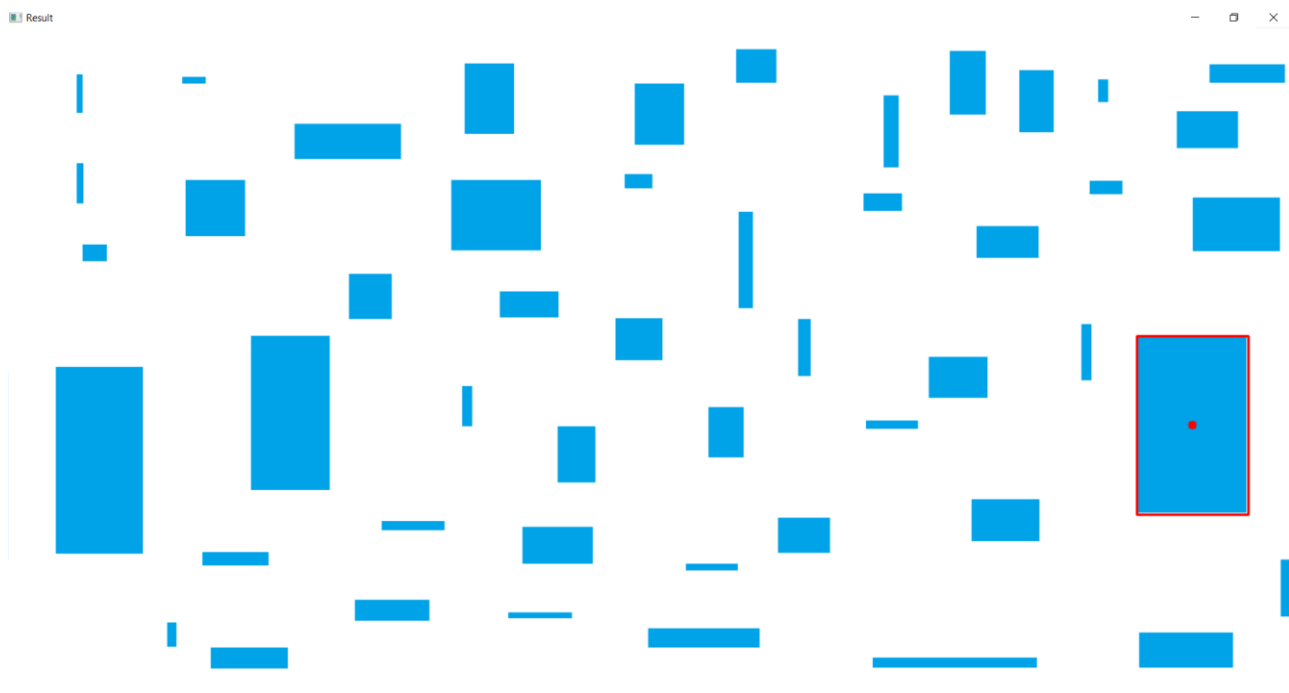


Рисунок 38 – Пример работы



Рисунок 39 – Пример работы

## **Итоги по экскурсии в компанию ISPsystem**

ISPsystem — небольшая технологическая компания, специализирующаяся на разработке платформ для комплексного управления виртуальной инфраструктурой. Ключевые направления включают создание программного обеспечения для управления физическим оборудованием, серверной виртуализацией и автоматизации учета и выдачи ИТ-ресурсов. По итогам посещения можно сказать, что компания стремится к постепенному росту за счёт наработанных стратегий или расширению старых.

Экосистема решений компании включает Dcmanager, VMmanager и billmanager. Dcmanager обеспечивает управление физической инфраструктурой и активами дата-центров; VMmanager отвечает за оркестрацию и администрирование виртуальных машин и ресурсов; billmanager поддерживает учет, биллинг и автоматизацию финансовых операций. Продукты логично сопряжены и обеспечивают сквозную автоматизацию жизненного цикла ИТ-ресурсов — от физического уровня до сервисного и финансового контуров. Рабочее пространство производит позитивное впечатление: интерьер выполнен в серо-черной гамме, современный, аккуратный и новый. Зоны организованы рационально; переговорные и общие пространства выделены, что способствует деловой коммуникации и концентрации. Визуальная сдержанность оформления поддерживает технологичный характер компании и общую дисциплину процессов.

Расположение офиса требует дополнительных затрат, однако такая удаленность компенсируется высокой степенью цифровизации внутренних процессов и готовностью команды к удаленным демонстрациям и обсуждениям. Коллектив производит благоприятное впечатление. Открытые к диалогу, доброжелательные. Высокая вовлеченность сотрудников позволила узнать много нового не заскучав ни на минуту. Общение со специалистами подтверждает тот уровень знаний в предметной области, к которому мы должны стремиться, а готовность оперативно разбирать вопросы и давать аргументированные ответы поддерживает динамику обмена знаниями.

Особые требования к сотрудникам наталкивают на осбую корпоративную дисциплину, что не может не радовать. Среди коллектива ISPsystem стоит атмосфера сплоченности и ответственности, какая доступна не многим.

## **Итоги по экскурсии в компанию АО «СО ЕЭС» Иркутское РДУ.**

АО «СО ЕЭС» — системный оператор Единой энергетической системы России, независимая государственная компания, организованная как отдельное предприятие с приоритетом обеспечения надежности работы энергосистемы, а не извлечения коммерческой выгоды. В структуре оператора действует 56 региональных диспетчерских управлений (РДУ). Иркутское РДУ является территориальным подразделением системного оператора и не относится к муниципальным структурам города Иркутска.

Профиль деятельности — оперативно-диспетчерское управление энергетикой региона с целью обеспечения устойчивого, бесперебойного и безопасного функционирования энергокомплекса. Ключевые задачи включают координацию работы генерирующих объектов и сетевой инфраструктуры, балансирование производства и потребления электроэнергии, обеспечение нормативных параметров режимов, а также управление потокораспределением и приоритизация поставок в штатных и нештатных ситуациях. Отдельный акцент сделан на действия при ЧП: от превентивного мониторинга и ситуационного анализа до оперативного принятия решений и восстановления нормальных режимов.

Иерархия диспетчерского управления трехуровневая: центральный уровень — центральное диспетчерское управление (ЦДУ) в Москве; субъектный уровень — семь Объединенных диспетчерских управлений (ОДУ), в периметр которых входит Иркутское РДУ (подчинение ОДУ Сибири, расположенному в Новосибирске); территориальный уровень — городские/региональные РДУ, непосредственно осуществляющие оперативное управление на местах. Такая вертикаль обеспечивает четкость и согласованность решений на всех ступенях управления.

ИТ-подразделение Иркутского РДУ выполняет небольшой круг функций, связанных в основном с инженерией: эксплуатация и обслуживание информационно-технологической инфраструктуры, обеспечение кибербезопасности и системное администрирование критически важных сервисов. Задачи включают поддержание высокой доступности и отказоустойчивости систем, контроль целостности и защищенности данных, своевременное обновление систем. ИТ-служба работает в тесной связке с диспетчерским блоком, обеспечивая непрерывную оперативную поддержку при технических изменениях и нештатных ситуациях.

В итоге компания практически не занимается ИТ отраслей. Основой для неё является управление энергетическими потоками их администрирование. В её задачи не входит автоматизация всех энергетических систем, настройка серверов или создание виртуальных сред для последующего управления.

## **Заключение**

В ходе учебной практики я смог расширить свои знания и навыки в программировании. Работа с семью задачами на C++ позволила мне улучшить алгоритмическое мышление и научиться писать более качественный код. Опыт с Arduino дал понять, как программирование связано с аппаратным обеспечением, а проект по распознаванию объектов на Python познакомил с основами машинного обучения.

Посещение IT-компаний "ISPsystem" и "АО «СО ЕЭС» Иркутское РДУ" позволило увидеть практическую работу в IT-сфере и пообщаться с профессионалами.

Эта практика помогла мне лучше понять свою специализацию и вдохновила на дальнейшее изучение и развитие в области информационных технологий.



## 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](https://alexgyver.ru/ws2812_guide)
3. AlexGyver: руководство по работе со светодиодными матрицами [Электронный ресурс]. – URL: [https://alexgyver.ru/matrix\\_guide](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>
7. Стуков И. Компьютерное зрение OpenCV: где применяется и как работает в Python [Электронный ресурс] / И. Стуков. — Skillbox, 2023. — URL: <https://skillbox.ru/media/code/kompyuternoe-zrenie-opencv-gde-primenyaetsya-i-kak-rabotaet-v-python/>

**Индивидуальное задание на прохождение**  
**учебной практики: технологической (проектно-технологической) практики**

для Симонова Александра Андреевича

(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: ИРНITU

Сроки прохождения практики с «16» июня 2023 г. по «29» июня 2025 г.

Цели и задачи прохождения практики:

Использование полученных в ходе изучения дисциплин «Информатика» и «Программирование» знаний в решении задач и оттачивание навыков написания эффективного кода.

Содержание практики, вопросы, подлежащие изучению:

Совершенствование навыков программирования на языке C++; освоение работы с микроконтроллерной платформой Arduino и изучение принципов взаимодействия с различными модулями этой системы; освоение основ программирования на языке Python; знакомство с базовыми приёмами машинного зрения

Планируемые результаты практики:

В ходе практики произошло углубление знаний языка программирования C++ и овладение навыками работы с Arduino. Сформированы умения разработки программного обеспечения для управления устройствами, получен опыт применения методов машинного зрения и развиты навыки отладки, тестирования и документирования программного кода.

Руководитель практики от  
института ИТиАД

Тко / Кононенко Р.В. /  
(подпись)

**Согласовано:**

Руководитель ООП

Тко / Кононенко Р.В. /  
(подпись)

« 16 » 06 2025 г.

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

Али «16» Июня 2025 г.  
(подпись)

## **ДНЕВНИК**

прохождения практики

обучающегося Симонова Александра Андреевича, ИСИБ-24-1  
(фамилия, имя, отчество, группа)

курс 1

направление Информатика и вычислительная техника

профиль Интеллектуальные системы обработки  
информации и управления

в ИРНИТУ

Иркутск 2025

Руководителем практики от структурного подразделения назначен:  
 Кононенко Роман Владимирович, доцент  
 (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	Выполнена задача №1, задача №2	Tko
2	17.06.2025	Выполнена задача №3	Tko
3	18.06.2025	Выполнена задача №4, задача №5, задача №6, составил резюме на hh.ru и superjor.ru	Tko
4	19.06.2025	Выполнена задача №7	Tko
5	20.06.2025	Выполнена задача №9, задача №10	Tko
6	22.06.2025	Выполнена задача №8	Tko
7	23.06.2025	Выполнена задача №11	Tko
8	27.06.2025	Заполнен отзыв на экскурсию в компанию ISPsystem. Заполнен отзыв на экскурсию в компанию АО «СО ЕЭС» Иркутское РДУ.	Tko

Дата фактического прибытия обучающегося в структурное подразделение 16.06.2025  
 Дата фактического убытия обучающегося из структурного подразделения 28.06.2025

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