

Министерство образования и науки РФ
Пермский национальный исследовательский политехнический университет
Электротехнический факультет
Кафедра информационных технологий и автоматизированных систем

Дискретная математика и математическая логика
Лабораторная работа № 7

Тема: «Алгоритм Шимбелла»

Выполнил: студент группы
ИВТ-23-1Б
Долганова Диана Евгеньевна
Проверил: ст. пр.
Рустамханова Г. И.

Оглавление

Цель работы	1
Задачи работы	2
Этапы выполнения	3
1. Чтение матрицы из файла	4
2. Печать матрицы	4
3. Функция Шимбелла	5
4. Главная функция	6
4.1 Настройка локали и чтение файла	6
4.2 Вывод матрицы	7
4.3 Вывод результата	7
Тестирование программы	8
Заключение	9

Цель работы

Найти максимум или минимум за определенное количество переходов с помощью алгоритма Шимбелла.

Задачи работы

1. На вход подается матрица 10×10 .
2. На выходе программа показывает максимум или минимум и количество переходов.

Этапы выполнения

Программа была разработана на языке C++ в программе среде Microsoft Visual Studio 2022.

1. Чтение матрицы из файла

Функция *get_matrix* принимает имя файла и считывает из него матрицу целых чисел. Она использует *ifstream* для чтения файла построчно и создает двумерный вектор, где каждая строка файла соответствуют одному вектору в матрице. После завершения чтения функция возвращает полученную матрицу (рис. 1).

```
vector<vector<int>> get_matrix(const string& file_name) {  
    ifstream file(file_name);  
    vector<vector<int>> matrix;  
    string line;  
  
    while (getline(file, line)) {  
        stringstream ss(line);  
        int value;  
        vector<int> row;  
        while (ss >> value) {  
            row.push_back(value);  
        }  
        matrix.push_back(row);  
    }  
  
    return matrix;  
}
```

Рисунок 1 - Функция *get_matrix*

2. Печать матрицы

Функция *print_matrix* (рис. 2) принимает матрицу в виде вектора векторов и выводит ее элементы на экран. Она перебирает строки и их элементы, выводя каждую строку на отдельной линии. В конце добавляется пустая строка для разделения выводов.

```

void print_matrix(const vector<vector<int>>& matrix) {
    for (const auto& row : matrix) {
        for (int elem : row) {
            cout << elem << " ";
        }
        cout << endl;
    }
    cout << endl;
}

```

Рисунок 2 - Функция print_matrix

3. Функция Шимбелла

Функция `shimbell` (рис. 3) реализует алгоритм Шимбелла для двухмерной матрицы. Она выполняет несколько шагов, в каждом из которых вычисляет новые значения для матрицы на основе текущих значений и выбранного режима (минимум или максимум). Функция возвращает итоговую матрицу после указанного количества шагов, учитывая все возможные комбинации значений:

- `int size = matrix.size();` — получает размер матрицы.
- `vector<vector<int>> result = matrix;` — инициализирует `result` той же матрицей.
- `for (int step = 0; step < steps - 1; ++step)` — цикл для выполнения алгоритма определенное количество шагов.
- `vector<vector<int>> new_result(size, vector<int>(size, 0));` — создает новую матрицу результат с нулями.
- Циклы для обхода по каждой ячейке матрицы.
- `vector<int> candidates;` — объявляет вектор для хранения возможных значений.
- Цикл `for (int k = 0; k < size; ++k)` — ищет подходящие элементы для текущей ячейки.
- `if (a != 0 && b != 0)` — проверяет, что оба элемента не равны нулю.

- `candidates.push_back(a + b);` — добавляет сумму к кандидатам.
- Проверяет, существуют ли кандидаты:
 - В случае "min" выбирает минимальный элемент.
 - В случае "max" выбирает максимальный элемент.
- Если кандидатов нет, устанавливает значение нуля.
- После завершения всех шагов возвращает итоговую матрицу.

```
vector<vector<int>> shimbell(const vector<vector<int>>& matrix, int steps, const string& mode) {
    int size = matrix.size();
    vector<vector<int>> result = matrix;

    for (int step = 0; step < steps - 1; ++step) {
        vector<vector<int>> new_result(size, vector<int>(size, 0));

        for (int i = 0; i < size; ++i) {
            for (int j = 0; j < size; ++j) {
                vector<int> candidates;

                for (int k = 0; k < size; ++k) {
                    int a = result[i][k];
                    int b = matrix[k][j];
                    if (a != 0 && b != 0) {
                        candidates.push_back(a + b);
                    }
                }

                if (!candidates.empty()) {
                    if (mode == "min") {
                        new_result[i][j] = *min_element(candidates.begin(), candidates.end());
                    }
                    else {
                        new_result[i][j] = *max_element(candidates.begin(), candidates.end());
                    }
                }
                else {
                    new_result[i][j] = 0;
                }
            }
        }
        result = new_result;
    }
    return result;
}
```

Рисунок 3 - Функция shimbell

4. Главная функция

4.1 Настройка локали и чтение файла

Добавляем строчку для корректного отображения русского языка и читаем файл с матрицей (рис. 4).

```

v int main() {
    setlocale(LC_ALL, "RU");
    string file_name = "g42.txt";
    auto matrix = get_matrix(file_name);

```

Рисунок 4 - Локаль и чтение файла

4.2 Вывод матрицы

Выводим матрицу, прочитанную из файла на экран (рис. 5).

```

cout << "Введена матрица:" << endl;
print_matrix(matrix);

```

Рисунок 5 - Вывод матрицы

4.3 Вывод результата

Запрашиваем режим работы: кратчайший или длиннейший путь и устанавливаем соответствующее значение. Запрашиваем количество шагов для алгоритма. Вызываем функцию Шимбелла и получаем результат. Выводим результат на экран (рис. 6) .

```

string mode;
int mode_code;
cout << "1. Кратчайший путь" << endl;
cout << "2. Длиннейший путь" << endl;
cin >> mode_code;
if (mode_code == 1) {
    mode = "min";
}
else if (mode_code == 2) {
    mode = "max";
}

int steps;
cout << "Введите количество переходов: ";
cin >> steps;

auto result = shimbell(matrix, steps, mode);

cout << "\nРезультат:" << endl;
print_matrix(result);

return 0;
}

```

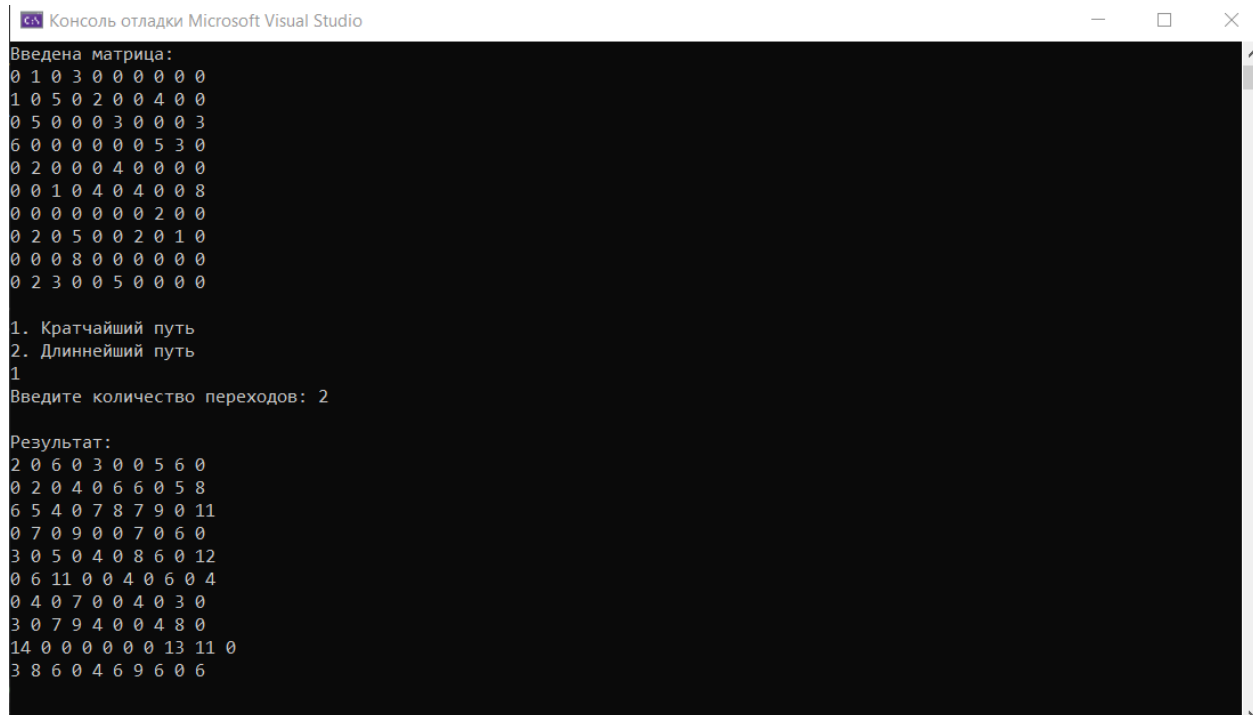
Рисунок 6 - Вывод результата

Тестирование программы

Для проверки программы на корректность добавим файл с матрицей и запустим программу (рис. 7 - 8).

0	1	0	3	0	0	0	0	0	0
1	0	5	0	2	0	0	4	0	0
0	5	0	0	0	3	0	0	0	3
6	0	0	0	0	0	0	5	3	0
0	2	0	0	0	4	0	0	0	0
0	0	1	0	4	0	4	0	0	8
0	0	0	0	0	0	0	2	0	0
0	2	0	5	0	0	2	0	1	0
0	0	0	8	0	0	0	0	0	0
0	2	3	0	0	5	0	0	0	0

Рисунок 7 - Первая матрица



```
Консоль отладки Microsoft Visual Studio
Введена матрица:
0 1 0 3 0 0 0 0 0 0
1 0 5 0 2 0 0 4 0 0
0 5 0 0 0 3 0 0 0 3
6 0 0 0 0 0 0 5 3 0
0 2 0 0 0 4 0 0 0 0
0 0 1 0 4 0 4 0 0 8
0 0 0 0 0 0 0 2 0 0
0 2 0 5 0 0 2 0 1 0
0 0 0 8 0 0 0 0 0 0
0 2 3 0 0 5 0 0 0 0

1. Кратчайший путь
2. Длиннейший путь
1
Введите количество переходов: 2

Результат:
2 0 6 0 3 0 0 5 6 0
0 2 0 4 0 6 6 0 5 8
6 5 4 0 7 8 7 9 0 11
0 7 0 9 0 0 7 0 6 0
3 0 5 0 4 0 8 6 0 12
0 6 11 0 0 4 0 6 0 4
0 4 0 7 0 0 4 0 3 0
3 0 7 9 4 0 0 4 8 0
14 0 0 0 0 0 0 13 11 0
3 8 6 0 4 6 9 6 0 6
```

Рисунок 8 - Результат

Заключение

Была разработана программа для нахождения минимума и максимума за определенное количество шагов по алгоритму Шимбелла.

Список используемой литературы

1. Алгоритм Шимбелла URL: <https://helpiks.org/5-58396.html> (дата обращения: 18.04.2025).
2. 8_topu_labor_rabota_8.pdf URL: https://gbpouberitt.ru/assets/8_topu_labor_rabota_8.pdf (дата обращения: 19.04.2025).