ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ**

**ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

(НИУ «БелГУ»)

ИНСТИТУТ ИНЖЕНЕРНЫХ И ЦИФРОВЫХ ТЕХНОЛОГИЙ

КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

**Отчет по лабораторной работе 3**

**по дисциплине: «Алгоритмы и анализ сложности»**

**Тема работы «Структуры данных для хранения графов»**

студента очного отделения

3 курса 12001801 группы

Капустина Виктора Сергеевича

Проверил(а):

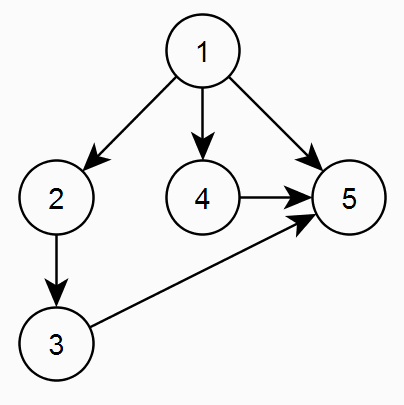
Гайворонский Виталий Александрович

Белгород 2021

**Цель работы:** Изучение основных структур данных, использующихся для хранения графов, и приобретение навыка разработки программ, преобразующих одни структуры в другие.

**Выполнение работы:** Требуется разработать алгоритм, преобразующий исходную матрицу смежности в список ребер.

Для примера был взят следующий граф:

  
Рисунок 1. Ориентированный граф

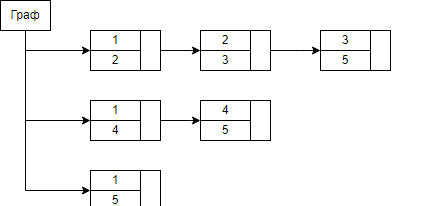
*Матрица смежности.* Матрица смежности представляет собой матрицу, где на пересечении вершин указывается наличие или отсутствие ребра (единицей и нулем соответственно) от x до y. Из построенной матрицы можно будет понять, что, например, от вершины под номером 1 идут ребра к вершинам под номерами 2, 4 и 5, так как на пересечении данных вершин будет стоять единица. Таким же образом работает определение существование ребер и между остальными вершинами.

Ниже представлена таблица – матрица смежности для приведенного выше графа. Рассмотрим матрицу смежности данного графа.

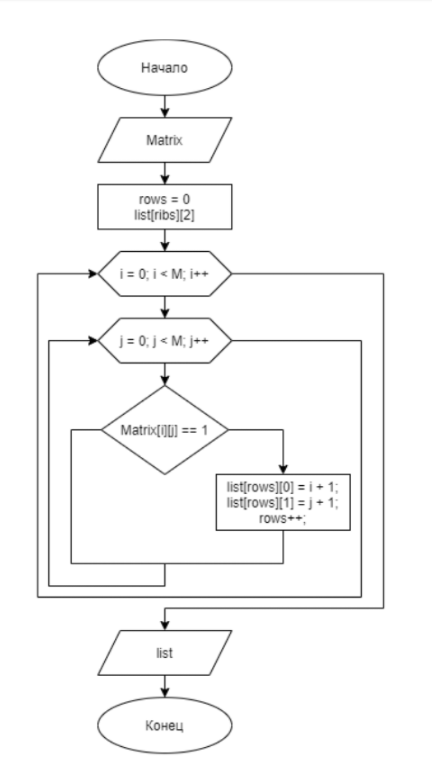
Таблица 1 – Матрица смежности

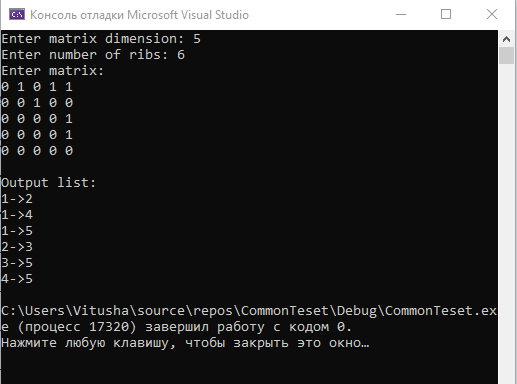
| x/y | 1 | 2 | 3 | 4 | 5 |
| --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 0 | 1 | 1 |
| 2 | 0 | 0 | 1 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 1 |
| 4 | 0 | 0 | 0 | 0 | 1 |
| 5 | 0 | 0 | 0 | 0 | 0 |

*Список ребер.* Список, в каждой строке которого записаны две связанные ребром вершины. Для ориентированного графа, первым значением будет то, из какой вершины исходит граф, а вторым, куда оно входит. По построенному списку ребер, будет видно, что каждая строка списка означает существующее ребро между указанными в этой же строке вершинами. Количество строк в списке должно соответствовать количеству ребер. Список ребер для взятого примера изображен на Рис. 2

  
Рисунок 2. Список ребер

*Алгоритм преобразования:* Для преобразования матрицы смежности в список ребер необходимо пройти по каждой строке матрицы смежности, при нахождении единицы, производить запись x и y в список ребер, где вершина из которой будет исходить ребро - x и вершина в которую будет входить ребро - y.

  
Рисунок 3. Блок-схема алгоритма преобразования

  
Рисунок 4. Результат работы программы

**Вывод:** Были изучены и рассмотрены две основные структуры данных, использующихся для хранения графов. Разобран пример построения ориентированного графа в каждой из описанных структур. Были приобретены навыки разработки программ, преобразующих одни структуры в другие.

**Листинг 1. Алгоритм преобразования**

void AdjacencyMatrixToListOfRibs(int\*\* matrix, int\*\* & list,int M) {

int rows = 0;

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

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

if (matrix[i][j] == 1) {

list[rows][0] = i + 1;

list[rows][1] = j + 1;

rows++;}}}}

void printList(int\*\* list, int rib) {

cout << "\nOutput list:\n";

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

cout << list[i][0] << "->" << list[i][1] << endl;;}}

int main(){

int M, rib;

cout << "Enter matrix dimension: ";

cin >> M;

cout << "Enter number of ribs: ";

cin >> rib;

int\*\* list;

list = new int\* [rib];

for (int i = 0; i < rib; i++)

list[i] = new int [2];

int\*\* matrix;

matrix = new int\* [M];

for (int i = 0; i < M; i++)

matrix[i] = new int[M];

cout << "Enter matrix:\n";

for (int i = 0; i < M; i++)

for (int j = 0; j < M; j++)

cin >> matrix[i][j];

AdjacencyMatrixToListOfRibs(matrix, list, M);

printList(list, rib);

return 0;}

**Конец листинга 1.**