Пензенский Государственный университет

Кафедра «Вычислительной техники»

Отчёт

По лабораторной работе №4

По дисциплине «Л и ОА в ИЗ»

На тему: «Обход графа в глубину»

***Выполнили студенты группы 19ВВ1:***

Даврешян А.

## *Приняли:*

## Митрохин М.А

## Пенза 2020

**Цель работы:**

1. Сгенерируйте (используя генератор случайных чисел) две матрицу смежности для неориентированного графа *G*. Выведите сгенерированные матрицы на экран.
2. Для сгенерированного графа осуществите процедуру обхода в глубину, реализованную в соответствии с приведенным выше описанием.

**Ход работы**

**Общие сведения.**

Обход графа – одна из наиболее распространенных операций с графами. Задачей обхода является прохождение всех вершин в графе. Обходы применяются для поиска информации, хранящейся в узлах графа, нахождения связей между вершинами или группами вершин и т.д.

Одним из способов обхода графов является поиск в глубину. Идея такого обхода состоит в том, чтобы начав обход из какой-либо вершины всегда переходить по первой встречающейся в процессе обхода связи в следующую вершину, пока существует такая возможность. Как только в процессе обхода исчерпаются возможности прохода, необходимо вернуться на один шаг назад и найти следующий вариант продвижения. Таким образом, итерационно выполняя описанные операции, будут пройдены все доступные для прохождения вершины. Чтобы не заходить повторно в уже пройденные вершины, необходимо их пометить как пройденные.

**Листинг**

#include "stdafx.h"

#include "iostream"

#include "conio.h"

#include "locale.h"

using namespace std;

int vis[10] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};

int \*\*matr;//указатель для массива указателей

void DFS (int s)

{

vis[s] = 1;

printf("%d ",s);

for(int i = 0; i < 10 ; i++)

{

if((matr[s][i] == 1) && (vis[i] == 0))

{

DFS(i);

}

}

}

int main()

{

int n,i,j;

cout << "Input matrix order:";

cin >> n;

matr = (int\*\*)malloc(n \* sizeof(int\*));

//выделение памяти под массив указателей

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

matr[i] = (int\*)malloc(n \* sizeof(int));

}

//выделение памяти для массива значений

for( i = 0; i < n; i++)

{

for( j = 0; j < n; j++)

{

if(i == j)

{

matr[i][j] = 0;

}

if (i < j)

{

matr[i][j] = rand()%2;

matr[j][i] = matr[i][j];

}

}

}

printf( "Result: ");

for(i = 0; i < n; i++)

{

cout << "\n";

for (j = 0; j < n; j++)

printf("%d\t",matr[i][j]);

}

printf( " \n ");

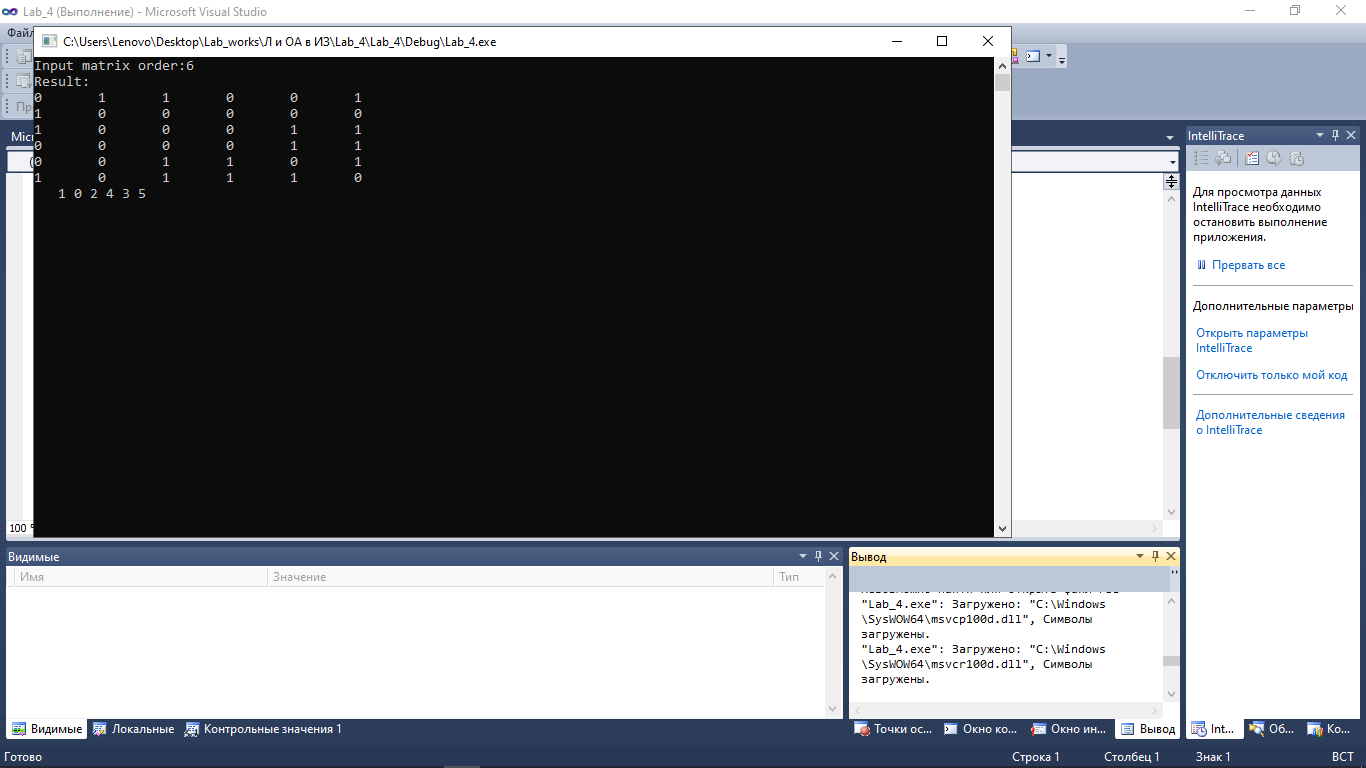
DFS(1);

getch();

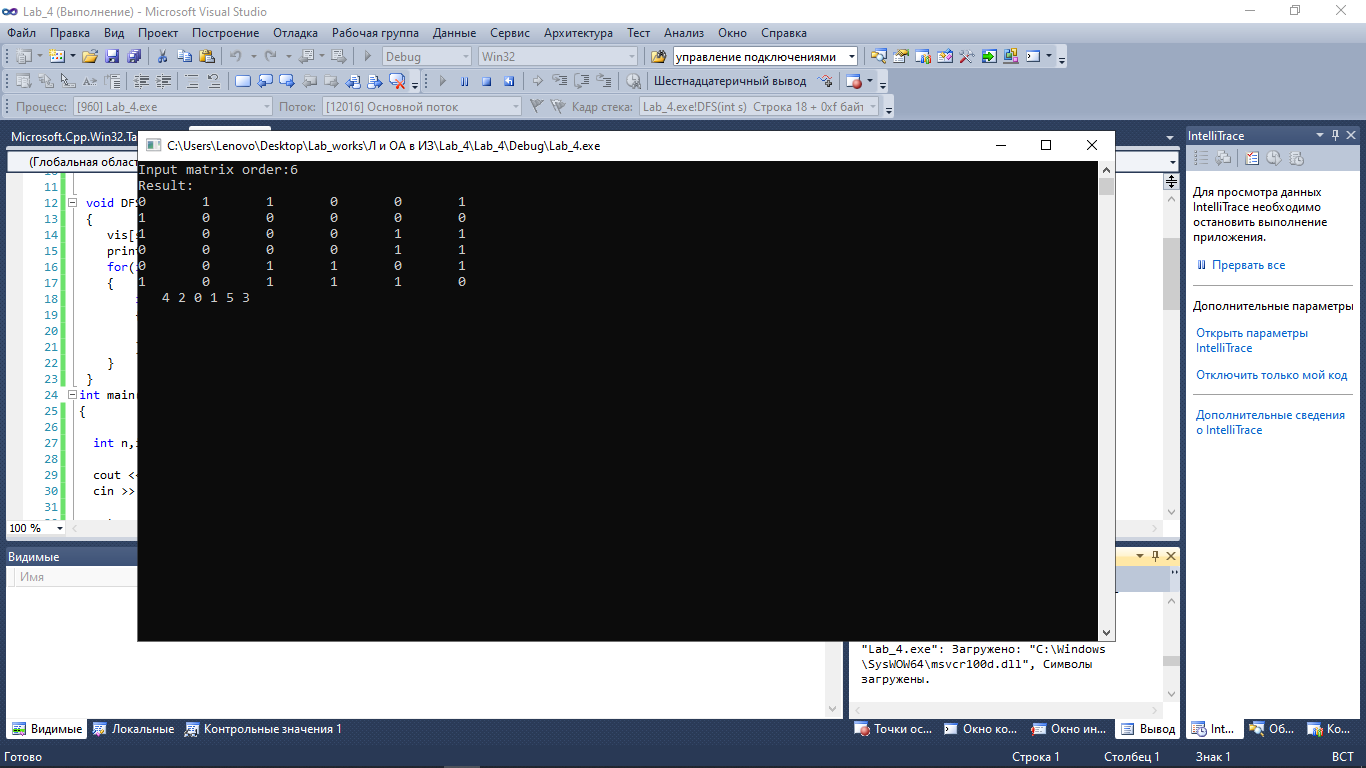
}

**Результат работы программы**

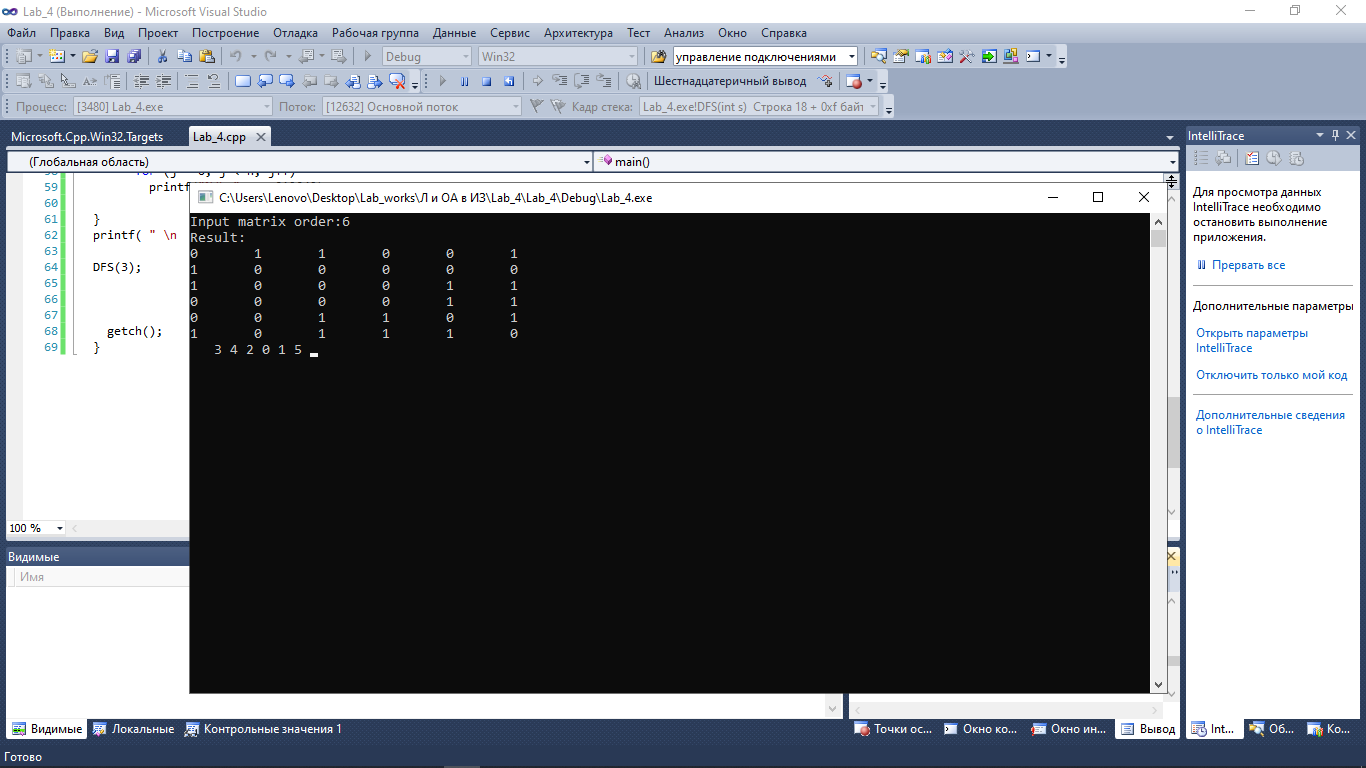
Результат обхода матрицы 6х6 с вершины 1



Результат обхода матрицы 6х6 с вершины 4



Результат обхода матрицы 6х6 с вершины 3



Результат трассировки программы

