Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №7

по курсу «Логика и основы алгоритмизации в инженерных задачах»

### на тему «Обход графа в глубину»

**Выполнили студенты**

**группы 21вв1.1:**

Федулов Е.Д.

Жбанников Д.Н.

**Приняли:**

Юрова О.В.

Акифьев И.В.

Пенза, 2022

**Цель работы:** научиться обходить граф в глубину, преобразовывать обход рекурсивной функции к нерекурсивной.

**Задания:**

### **Задание 1**

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

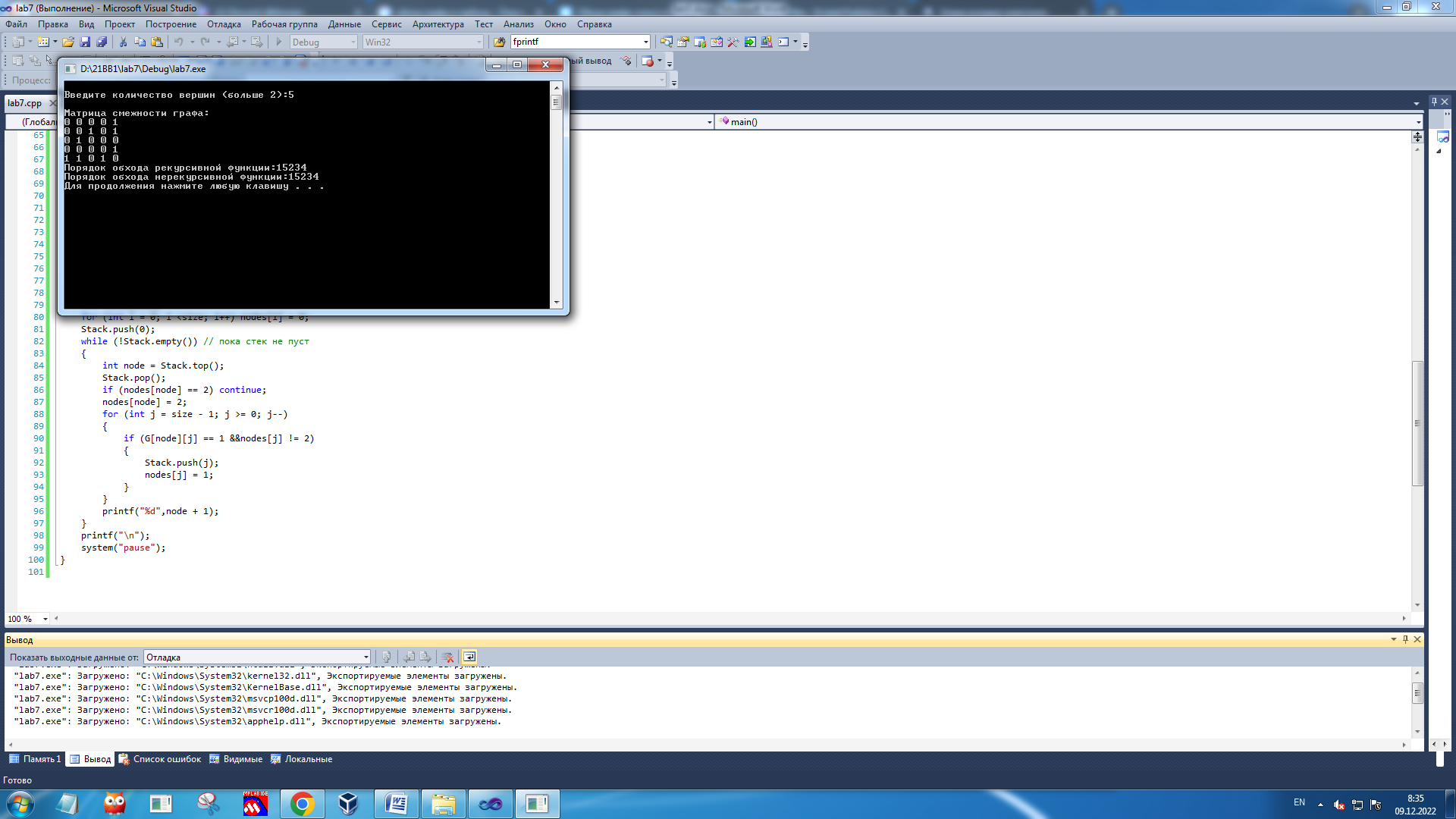
**3.**\* Реализуйте процедуру обхода в глубину для графа, представленного списками смежности.

### **Задание 2\***

1. Для матричной формы представления графов выполните преобразование рекурсивной реализации обхода графа к не рекурсивной.

**Ход работы**

**Результат программы для основного и дополнительного задания:**

****

**Вывод:** научились обходить граф в глубину, преобразовывать обход рекурсивной функции к нерекурсивной.

**Общий листинг:**

#include<stack>

#include<string>

#include<stdlib.h>

#include<stdio.h>

#include<limits>

#include<time.h>

using namespace std;

int i, j,size;

bool \*visited=new bool[200];

//матрица смежности

int \*\*G;

int\*\* M(int size){

int \*\*G = (int\*\*)malloc(sizeof(int\*)\*size);

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

G[i] = (int\*)malloc(sizeof(int)\*size);

}

return G;

}

int\*\* init(int\*\* graph, int size){

srand(time(NULL));

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

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

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

}

}

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

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

if(i == j){

G[i][j] = 0;

}

if (G[i][j] = G[j][i]){

G[i][j] = G[j][i];

}

}

}

return G;

}

//поиск в глубину

void DFS(int st)

{

int r;

printf("%d",st+1);

visited[st]=true;

for (r=0; r<=size; r++)

if ((G[st][r]!=0) && (!visited[r]))

DFS(r);

}

void main()

{

setlocale(LC\_ALL, "Rus");

int start = 1;

printf("\nВведите количество вершин (больше 2):");

scanf("%d", &size);

G = M(size);

init(G,size);

printf("\nМатрица смежности графа:");

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

printf("\n");

visited[i]=false;

for (j=0; j<size; j++){

printf("%d ",G[i][j]);

}

}

bool \*vis=new bool[size];

printf("\nПорядок обхода рекурсивной функции:");

DFS(start-1);

delete []visited;

printf("\nПорядок обхода нерекурсивной функции:");

stack<int>Stack;

int nodes[200];

for (int i = 0; i <size; i++) nodes[i] = 0;

Stack.push(0);

while (!Stack.empty()) // пока стек не пуст

{

int node = Stack.top();

Stack.pop();

if (nodes[node] == 2) continue;

nodes[node] = 2;

for (int j = size - 1; j >= 0; j--)

{

if (G[node][j] == 1 &&nodes[j] != 2)

{

Stack.push(j);

nodes[j] = 1;

}

}

printf("%d",node + 1);

}

printf("\n");

system("pause");

}