Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №4

по курсу «Л и ОА в ИЗ»

на тему «Простые структуры данных»

**Выполнили:**

студентом группы 19ВВ3

Субботкин М.В.

**Принял:**

Митрохин М.А.

Пенза 2020

**Цель работы:**

Изучить простые структуры данных.

**Ход работы:**

**Задание 1**

**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.**

**Задание 2\***

**1. Для матричной формы представления графов выполните**

**преобразование рекурсивной реализации обхода графа к не рекурсивной.**

**Листинг:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <stack>

#include <queue>

using namespace std;

void DFS(int vert, int\* vis, int\*\* mass, int f) {

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

vis[vert] = 1;

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

if (mass[vert][i] && vis[i] != 1) {

DFS(i, vis, mass, f);

}

}

}

void norecurses(int vertex, int\*\* mass, int\* vis, int f) {

stack<int>DFSstack;

DFSstack.push (vertex);

vis[vertex] = 1;

while (!DFSstack.empty()) {

vertex = DFSstack.top();

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

DFSstack.pop();

for (int i = f ; i >= 0; i--) {

if (mass[vertex][i] == 1 && vis[i]!= 1) {

vis[i] = 1;

DFSstack.push(i);

}

}

}

}

int main() {

srand(time(NULL));

int f;

printf("Enter lenght: ");

scanf("%d", &f);

int\* vis = (int\*)malloc(sizeof(int) \* f);

int\* vert = (int\*)malloc(sizeof(int) \* f);

int\*\* mass = (int\*\*)malloc(sizeof(int) \* f);

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

vis[i] = 0;

}

printf("\n");

printf("\n");

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

vert[i] = i + 1;

printf("%d ", vert[i]);

}

printf("\n");

printf("\n");

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

mass[i] = (int\*)malloc(sizeof(int) \* f);

}

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

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

int c = rand() % 101;

if (c <= 50) {

mass[i][j] = 1;

}

else mass[i][j] = 0;

if (i == j) {

mass[i][j] = 0;

}

}

}

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

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

if (mass[i][j] != mass[j][i])

mass[i][j] = mass[j][i];

}

}

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

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

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

}

printf("\n");

}

for (int vert = 0; vert < f; vert++) {

if (vis[vert] == 0) {

printf("\n");

DFS(vert, vis, mass, f);

}

}

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

vis[i] = 0;

}

printf("\n");

for (int vert = 0; vert < f; vert++) {

if (vis[vert] == 0) {

printf("\n");

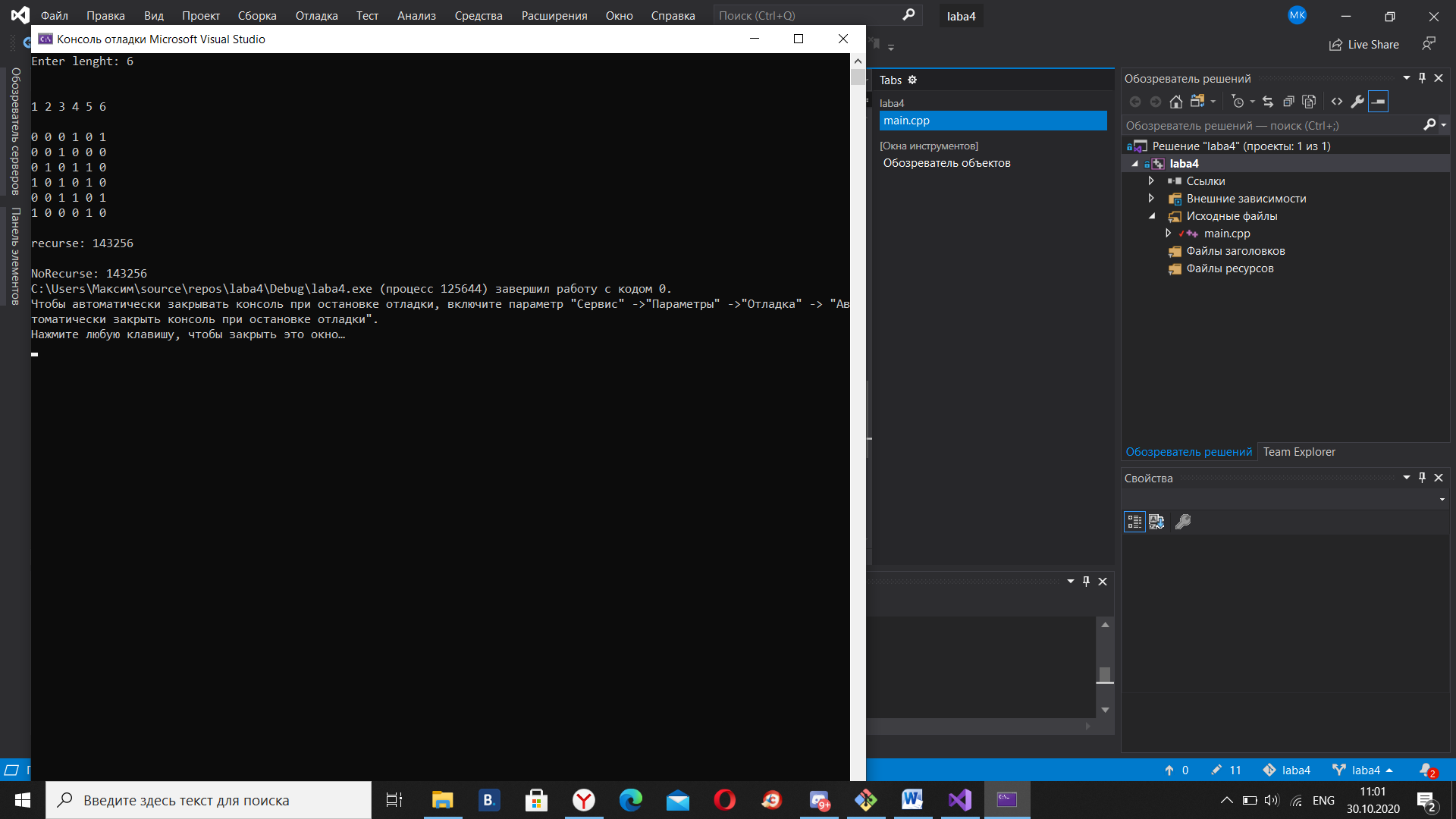
norecurses(vert, mass, vis, f);

}

}

}

### Результат работы программы:



### Выводы

В ходе выполнения лабораторной работы мы изучили и реализовали обход в глубину графа.