Пензенский государственный университет  
Кафедра «Вычислительной техники»

**Отчет**по лабораторной работе №8  
по дисциплине: «Логика и основы алгоритмизации в инженерных задачах»  
на тему: «Определение характеристик графов»

**Выполнил студент группы 19ВВ3:**

Филатова Д.С.

**Принял:**

Митрохин М. А.

Пенза 2020.

**Цель работы:** научится находить эксцентриситет вершины, определять радиус и диаметр графа, находить изолированные, концевые и доминирующие вершины.

**Листинг:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <locale>

#include <iostream>

#include <conio.h>

#include <time.h>

#include <stack>

#include <queue>

#include <stdio.h>

#include <Windows.h>

#include <stdlib.h>

using namespace std;

int\* dist;

int\* dist\_2;

int\* visited;

int\*\* graph; // взвешанный ориентированный

int\*\* graph\_2; // взвешанный неориентированный

queue<int>Q;

queue<int>QQ;

int start, start\_2;

int N, N\_2, j, i, R, max1, dia;

int\* ecs;

void BFSD(int v, int N, int\*\* graph) {

Q.push(v);

dist[v] = 0;

while (!Q.empty()) {

v = Q.front();

Q.pop();

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

if ((graph[v][i] > 0) && (dist[i] == -1)) {

Q.push(i);

dist[i] = dist[v] + graph[v][i];

}

}

}

}

int main() {

srand(time(NULL));

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

printf("Ведите размер матрицы ");

scanf\_s("%d", &N\_2);

srand(time(NULL));

graph\_2 = new int\* [N\_2];

dist\_2 = new int[N\_2];

visited = new int[N\_2];

ecs = new int[N\_2];

for (int i = 0; i < N\_2; i++)

{

graph\_2[i] = new int[N\_2];

}

for (int i = 0; i < N\_2; i++)

{

dist\_2[i] = -1;

}

for (int i = 0; i < N\_2; i++)

{

ecs[i] = 0;

}

printf("\n Матрица смежности для взвешенного неориентированного графа \n\n");

for (i = 0; i < N\_2; i++)

{

for (j = i; j < N\_2; ++j)

{

graph\_2[i][j] = graph\_2[j][i] = rand() % 10;

graph\_2[i][i] = graph\_2[j][j] = 0; // чтобы петля(узел) не создавалась

}

}

printf(" ");

for (j = 0; j < N\_2; j++)

{

printf("%4d ", j + 1); //горизонт

}

printf("\n\n");

for (i = 0; i < N\_2; i++) {

printf(" %d ", i + 1); // вертик

for (j = 0; j < N\_2; j++)

{

printf("%4d ", graph\_2[i][j]);

}

printf("\n\n");

}

printf("\n\n Введите начальную вершину ");

for (start\_2 = 0; start\_2 < N\_2; start\_2++) {

//BFSD\_2(start\_2 - 1, N\_2, graph\_2);

dist = new int[N\_2];

for (int i = 0; i < N\_2; i++)

{

dist[i] = -1;

}

BFSD(start\_2, N\_2, graph\_2);

printf("\n\n Вершины ");

for (int i = 1; i <= N\_2; i++) {

printf("|%3d ", i);

}

printf("\n Расстояние ");

for (int i = 0; i < N\_2; i++) {

printf("|%3d ", dist[i]);

}

printf("\n\n");

max1 = 0;

for (int i = 0; i < N\_2; i++)

{

if (dist[i] >= max1)

max1 = dist[i];

}

ecs[start\_2] = max1;

{

dia = 0;

for (int i = 0; i < N\_2; i++)

{

if (ecs[i] >= dia)

dia = ecs[i];

}

ecs[start\_2] = dia;

}

}

printf("\n эксцентриситет ");

for (int i = 0; i < N\_2; i++) {

printf("|%3d ", ecs[i]);

}

for (int i = 0; i < N\_2; i++)

{

if (dist[i] >= max1)

max1 = dist[i];

}

ecs[start\_2] = max1;

printf("\n диаметр ");

max1 = 0;

for (int i = 0; i < N\_2; i++)

{

if (ecs[i] >= max1)

max1 = ecs[i];

}

printf("%d", max1);

printf("\n радиус ");

int min = 1000000;

for (int i = 0; i < N\_2; i++)

{

if (ecs[i] <= min) {

if (ecs[i]) {

min = ecs[i];

}

}

}

printf("%d", min);

printf("\n центральная вершина ");

for (int i = 0; i < N\_2; i++)

{

if (ecs[i] == min) {

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

}

}

printf("\n переферийная вершина ");

for (int i = 0; i < N\_2; i++)

{

if (ecs[i] == max1) {

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

}

}

printf("\n изолированная вершина ");

for (int i = 0; i < N\_2; i++)

{

if (ecs[i] == 0) {

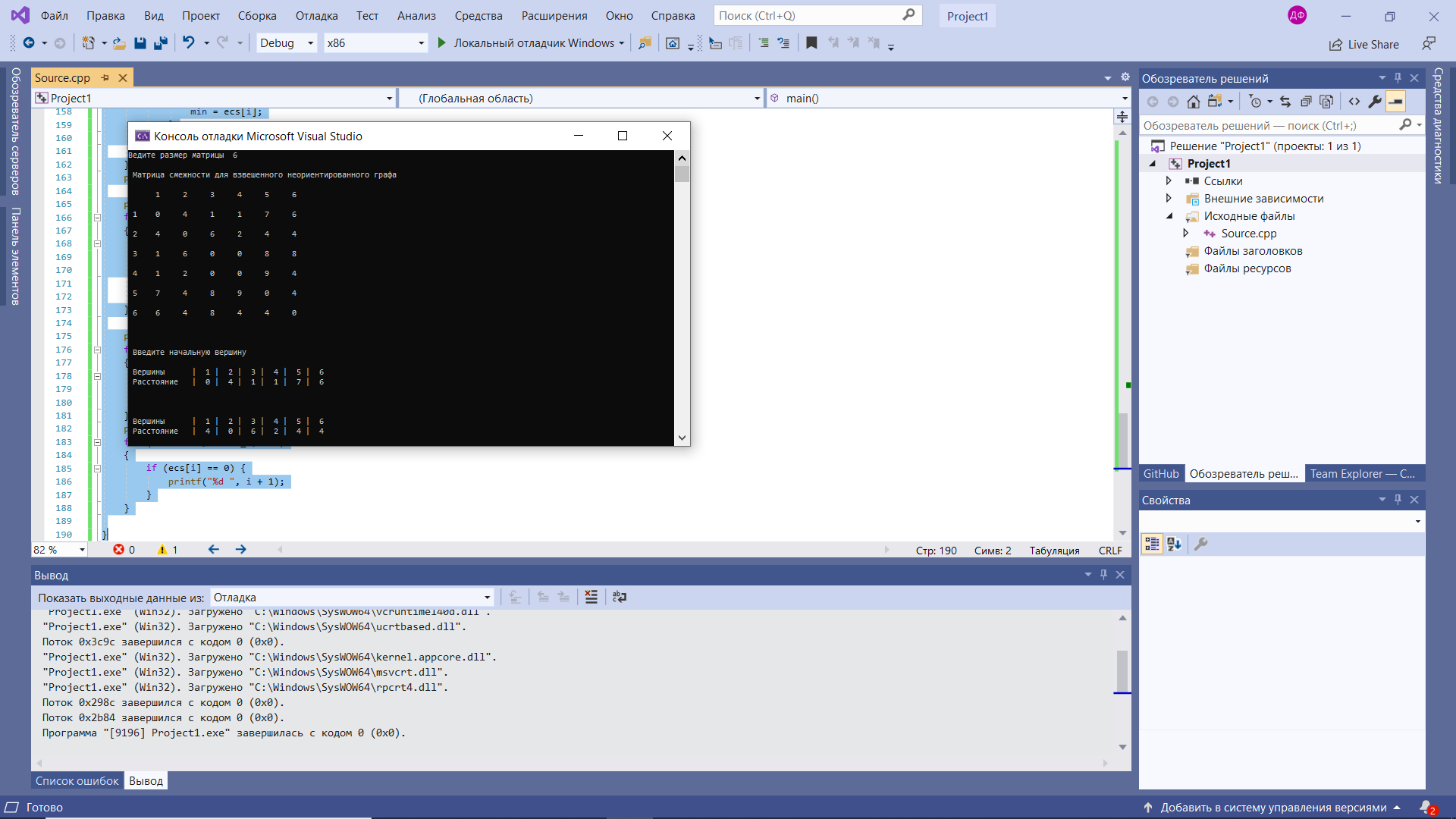
printf("%d ", i + 1);

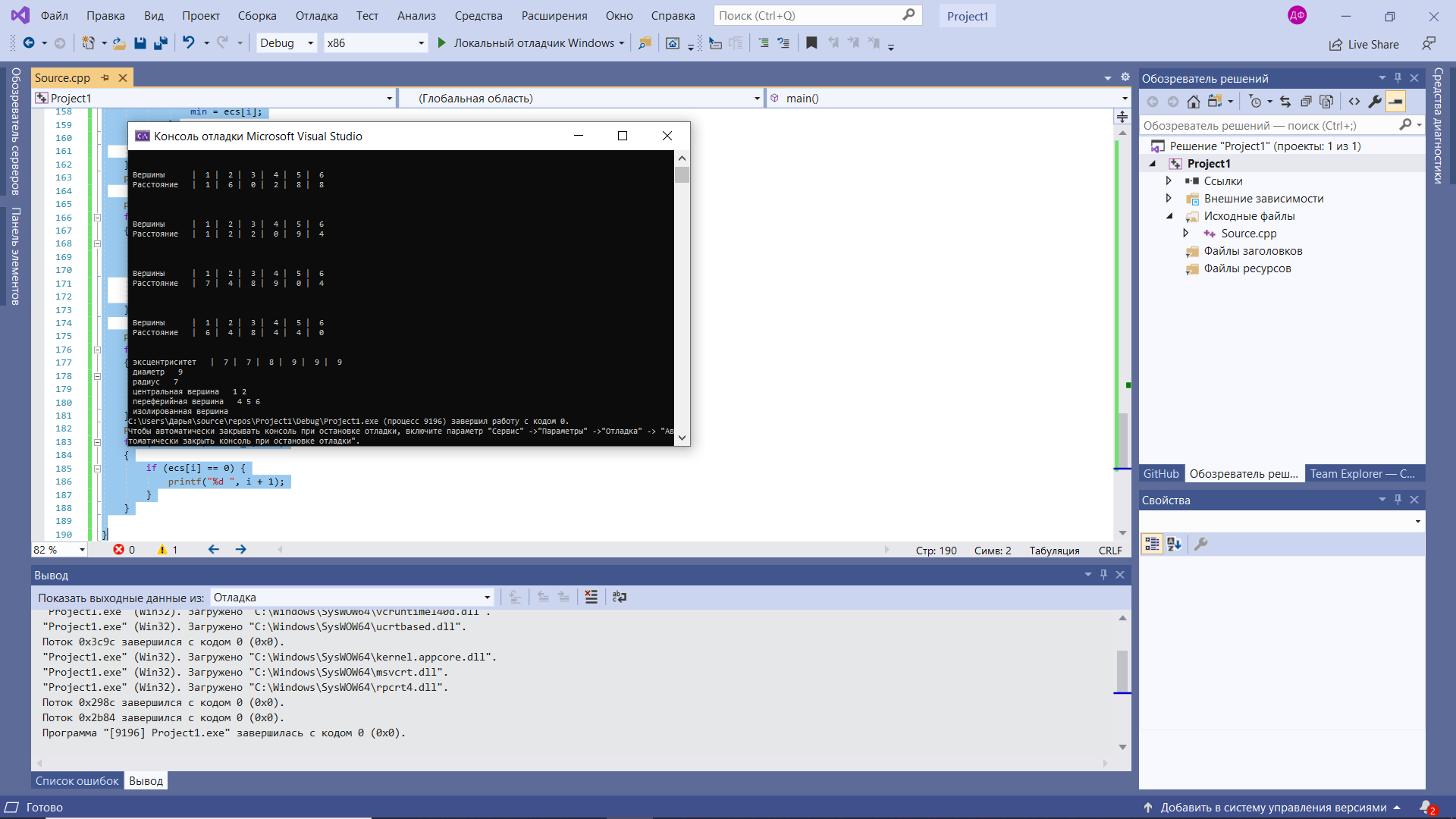
}

}

}

**Результат работы кода**





**Вывод:** научился находить эксцентриситет вершины, определять радиус и диаметр графа, находить изолированные, концевые и доминирующие вершины.