Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №8

**по курсу «Логика и основы алгоритмизации в инженерных задачах»**

на тему «Определение характеристик графов»

Выполнили:

студент группы 20ВВ2

Сурков Максим Сергеевич

Принял:

к.т.н., доцент

Митрохин М.А.

**Пенза 2021**

**Цель работы:** реализовать программу для нахождения характеристик графа.

**Лабораторные задания:**

Задание 1

Сгенерируйте (используя генератор случайных чисел) матрицу

смежности для неориентированного взвешенного графа G. Выведите матрицу

на экран.

2. Определите радиус и диаметр графа G, используя матрицу смежности

графа.

3. Определите подмножества периферийных и центральных вершин

графа G, используя матрицу смежности.

4. Найдите изолированные, концевые и доминирующие вершины.

Задание 2\*

1. Постройте для графа G матрицу инцидентности.

2. Определите радиус и диаметр графа G, используя матрицу

инцидентности графа.

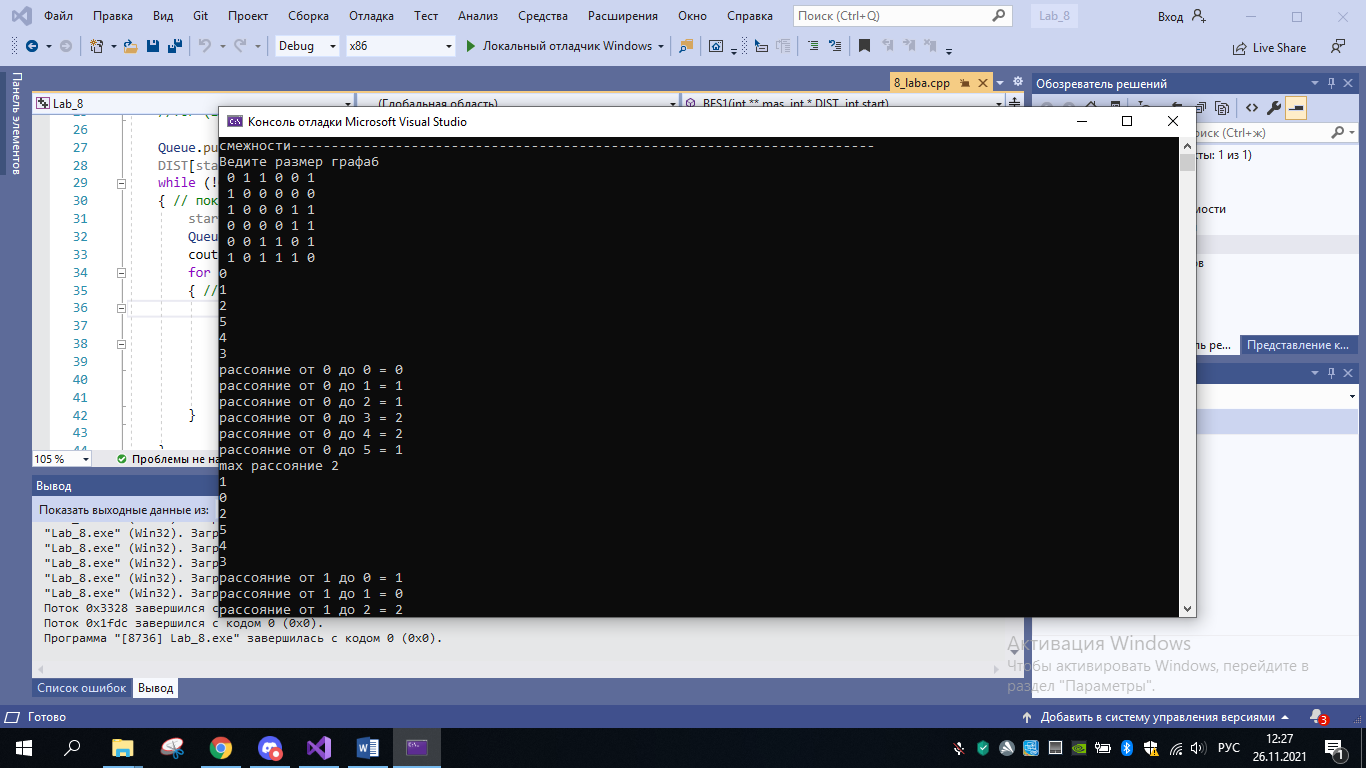
3. Определите подмножества периферийных и центральных вершин

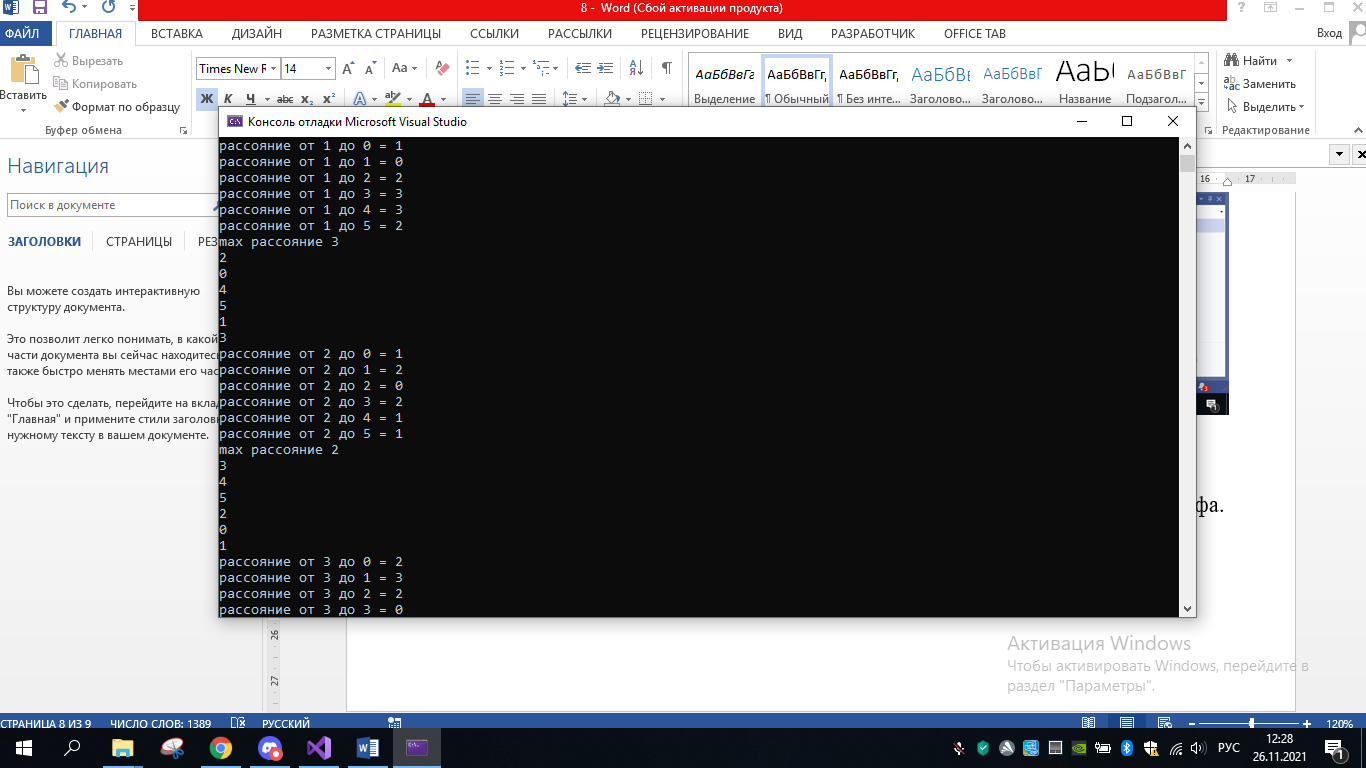
графа G, используя матрицу инцидентности.

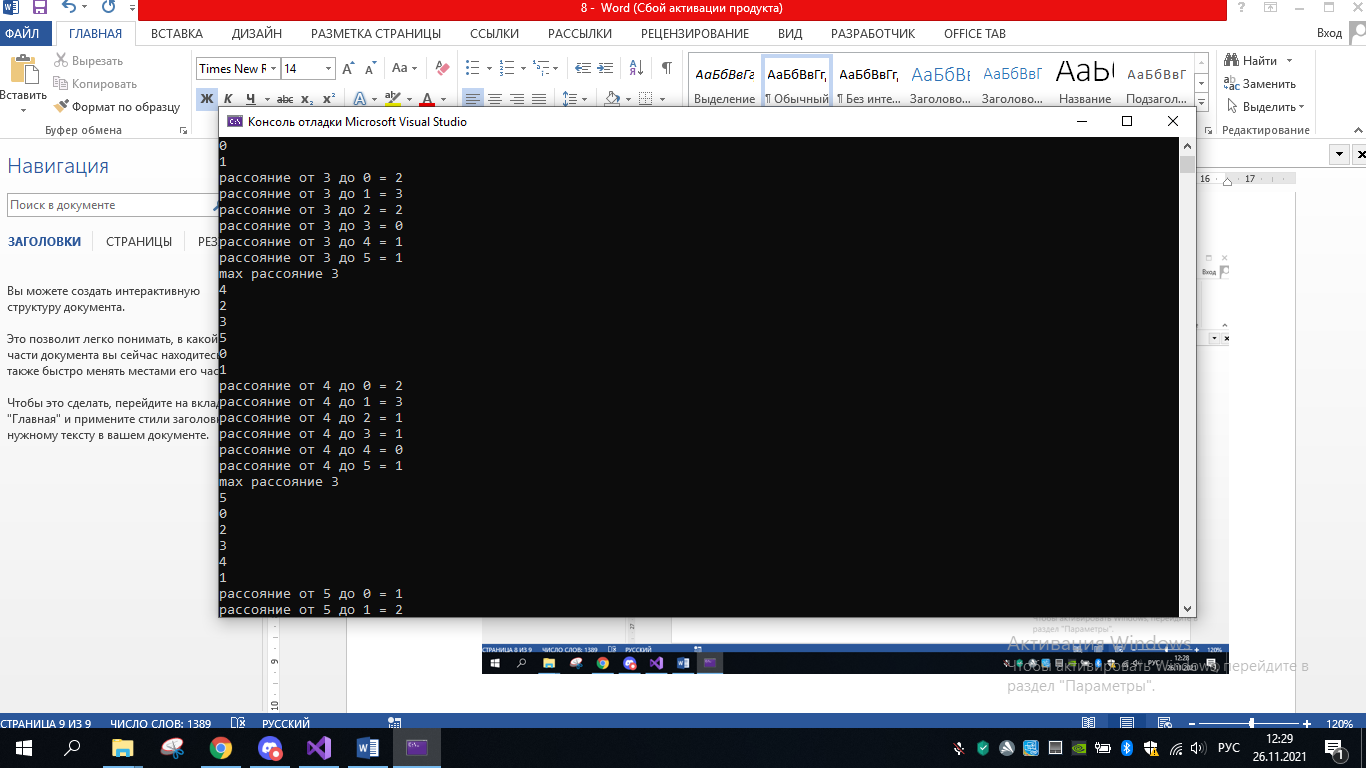
**Листинг программы:**

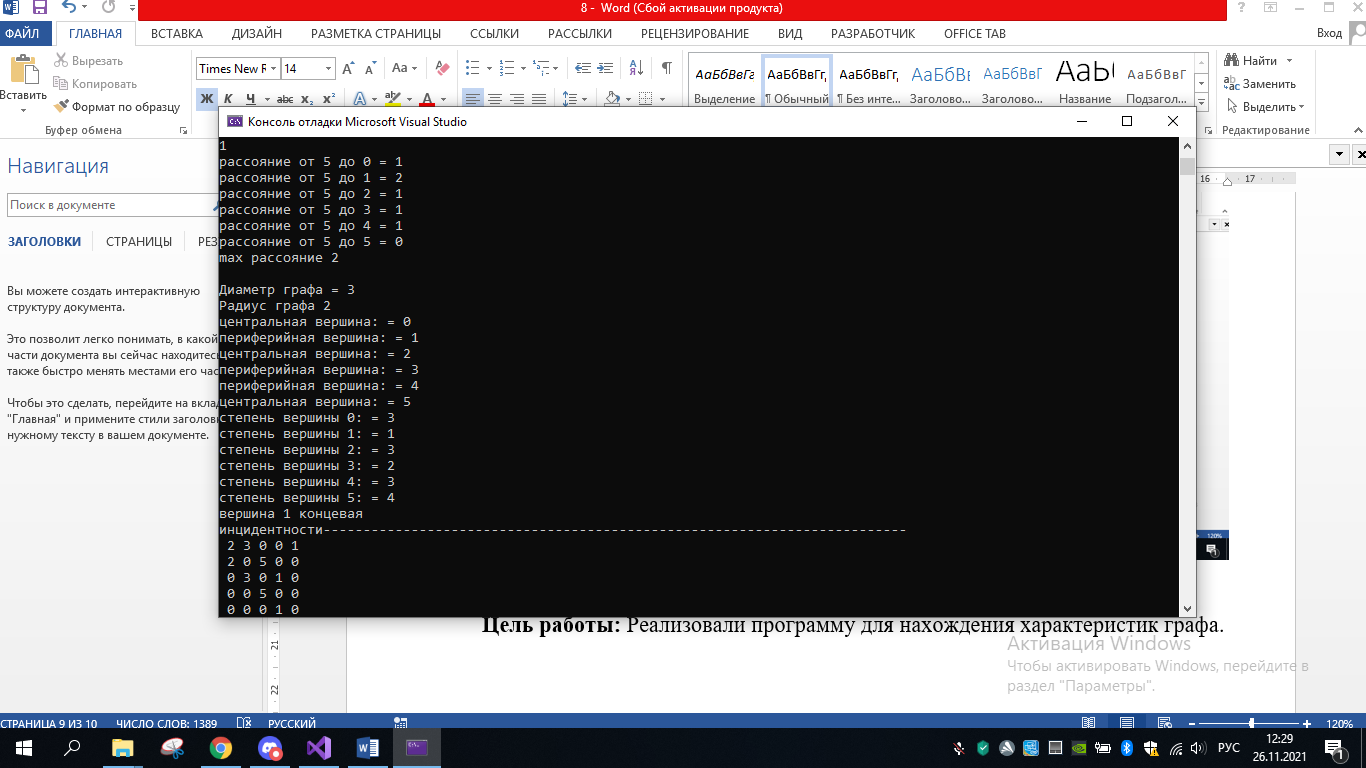
[#include](https://vk.com/im?sel=176176983&st=%23include) <iostream>  
[#include](https://vk.com/im?sel=176176983&st=%23include) <stdio.h>  
[#include](https://vk.com/im?sel=176176983&st=%23include) <stdlib.h>  
[#include](https://vk.com/im?sel=176176983&st=%23include) <cstdlib>  
[#include](https://vk.com/im?sel=176176983&st=%23include) <windows.h>  
[#include](https://vk.com/im?sel=176176983&st=%23include) <conio.h>  
[#include](https://vk.com/im?sel=176176983&st=%23include) <ctime>  
[#include](https://vk.com/im?sel=176176983&st=%23include) <cmath>  
[#include](https://vk.com/im?sel=176176983&st=%23include) <stack>  
[#include](https://vk.com/im?sel=176176983&st=%23include) <queue>  
  
  
using namespace std;  
  
int max\_size = 7, i, j, start, num, mass\_ex[30], D=0, R=1000, perif, center, elem=0;  
int\* DIST = (int\*)malloc(max\_size \* sizeof(int));  
queue<int> ex;  
  
void BSFD(int st, int\*\* b, int num\_of\_elem, int\* vis) {  
queue<int> Q;  
int t;  
Q.push(st);  
vis[st] = 0;  
  
while (!Q.empty())  
{  
t = Q.front();  
Q.pop();  
for (int i = 0; i < num\_of\_elem; i++)  
{  
if (b[t][i] > 0 && vis[i] == 1000)  
{  
vis[i] = vis[t] + b[t][i];  
Q.push(i);  
}  
}  
}  
}  
  
void max\_elem(int\* mass){  
int max=0, n;  
for (n = 0; n < max\_size; n++) {  
if (mass[n] > max and mass[n] != 1000)  
max = mass[n];  
}  
ex.push(max);  
}  
  
int main() {  
  
setlocale(LC\_ALL, "Rus");  
srand(time(NULL));  
  
int\*\* a = new int\* [max\_size];  
for (i = 0; i < max\_size; i++)  
{  
a[i] = new int[max\_size];  
}  
  
for (i = 0; i < max\_size; i++)  
{  
for (j = i + 1; j < max\_size; j++)  
{  
num = rand() % 8;  
if (num <= 4)  
{  
a[i][j] = 0;  
a[j][i] = a[i][j];  
}  
else  
{  
a[i][j] = (rand() % 6) + 1;  
a[j][i] = a[i][j];  
}  
}  
}  
printf("\nНеориентированный граф:\n");  
for (i = 0; i < max\_size; i++)  
{  
DIST[i] = 1000;  
for (j = 0; j < max\_size; j++)  
{  
if (i == j)  
a[i][j] = 0;  
printf("%2d", a[i][j]);  
}  
printf("\n");  
}  
  
for (i = 0; i < max\_size; i++) {  
BSFD(i, a, max\_size, DIST);  
max\_elem(DIST);  
printf("\nРасстояния из %d-й вершины: ", i + 1);  
for (j = 0; j < max\_size; j++) {  
printf("%4d ", DIST[j]);  
DIST[j] = 1000;  
}  
}  
  
printf("\n\nЭксицентритеты вершин:\n");  
for (i = 0; i < max\_size; i++) {  
if (!ex.empty()) {  
mass\_ex[i] = ex.front();  
printf("%d-й = %d\n", i + 1, mass\_ex[i]);  
ex.pop();  
}  
}  
  
for (i = 0; i < max\_size; i++) {  
if (mass\_ex[i] > D)  
D = mass\_ex[i];  
if (mass\_ex[i] < R)  
R = mass\_ex[i];  
}  
printf("Диаметр графа = %d\nРадиус графа = %d", D, R);  
  
printf("\nПерифирийные вершины: ");  
for (i = 0; i < max\_size; i++) {  
if (mass\_ex[i] == D)  
printf("%d ", i+1);  
}  
printf("\nЦентральные вершины: ");  
for (i = 0; i < max\_size; i++) {  
if (mass\_ex[i] == R)  
printf("%d ", i+1);  
}  
  
//изолированные, концевые и доминирующие вершины  
printf("\n\nКонцевые вершины:");  
for (i = 0; i < max\_size; i++)  
{  
for (j = 0; j < max\_size; j++)  
{  
if (a[i][j] > 0)  
elem++;  
}  
if (elem == 1)  
printf("%d ", i + 1);  
elem = 0;  
}  
  
printf("\nИзолированные вершины:");  
for (i = 0; i < max\_size; i++)  
{  
for (j = 0; j < max\_size; j++)  
{  
if (a[i][j] > 0)  
elem++;  
}  
if (elem == 0)  
printf("%d ", i + 1);  
elem = 0;  
}  
  
printf("\nДоминирующие вершины:");  
for (i = 0; i < max\_size; i++)  
{  
for (j = 0; j < max\_size; j++)  
{  
if (a[i][j] > 0)  
elem++;  
}  
if (elem == max\_size - 1)  
printf("%d ", i + 1);  
elem = 0;  
}  
}

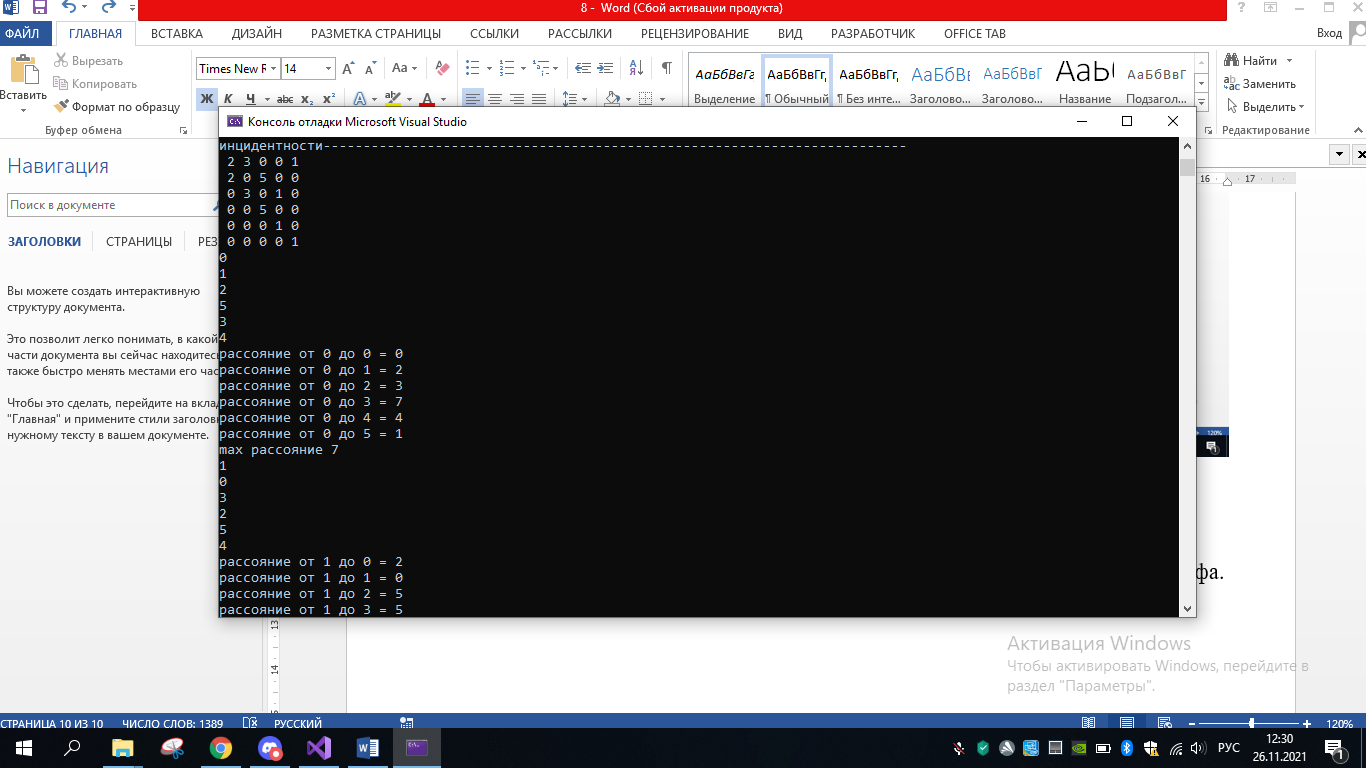
**Результаты работы программы:**

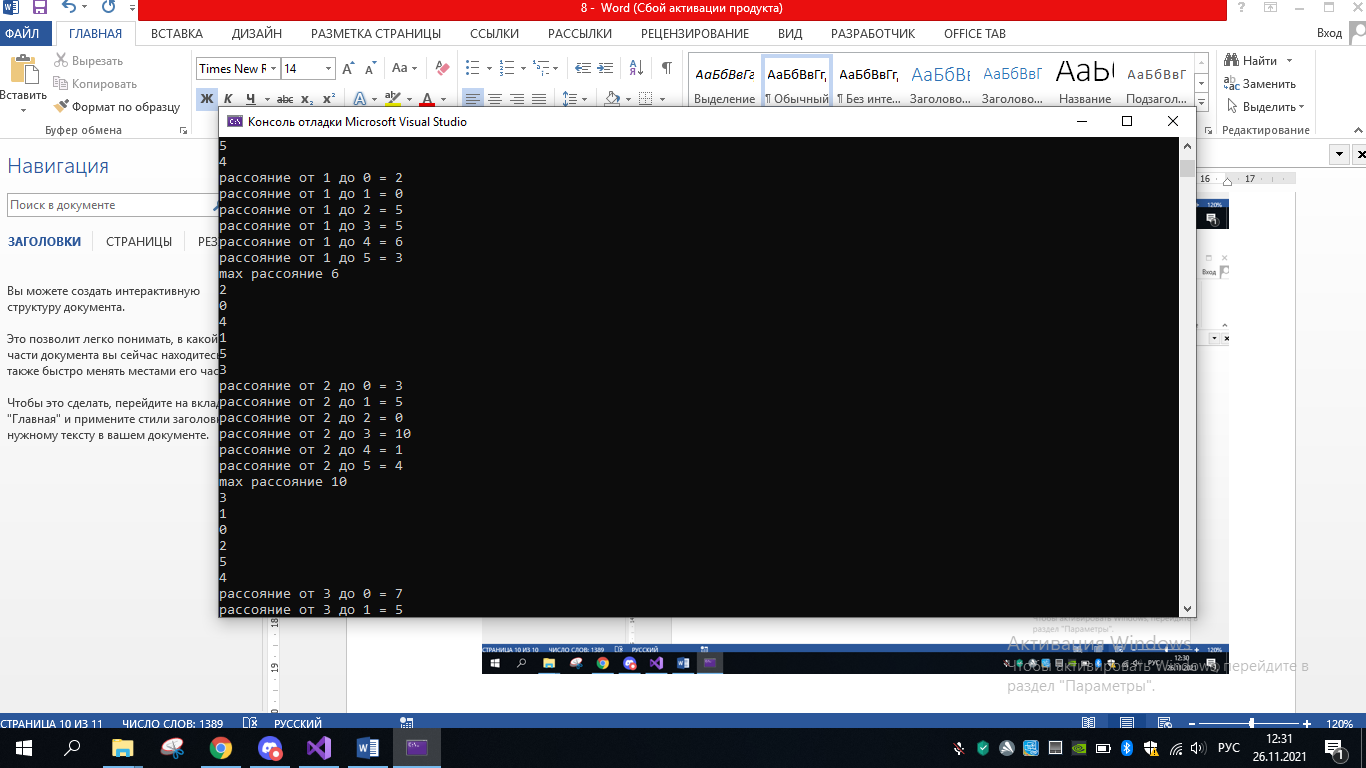


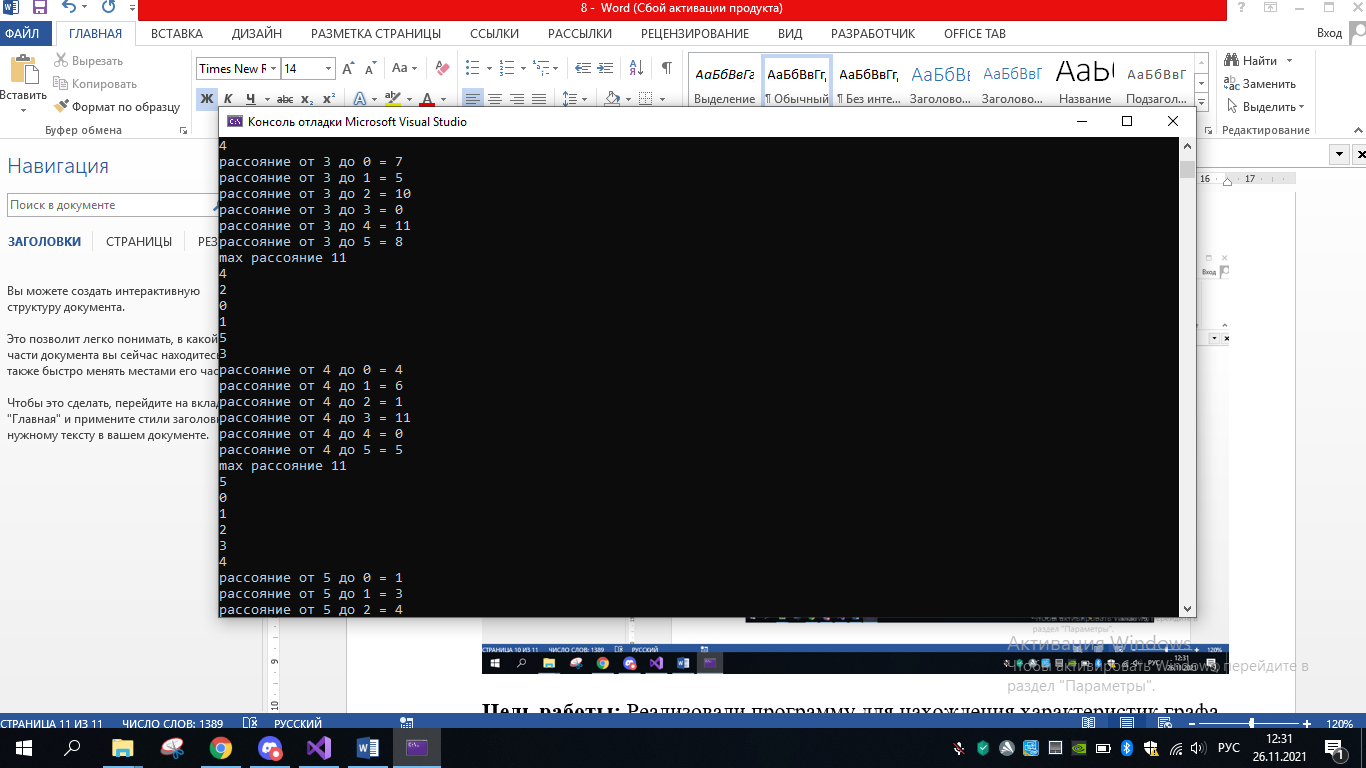


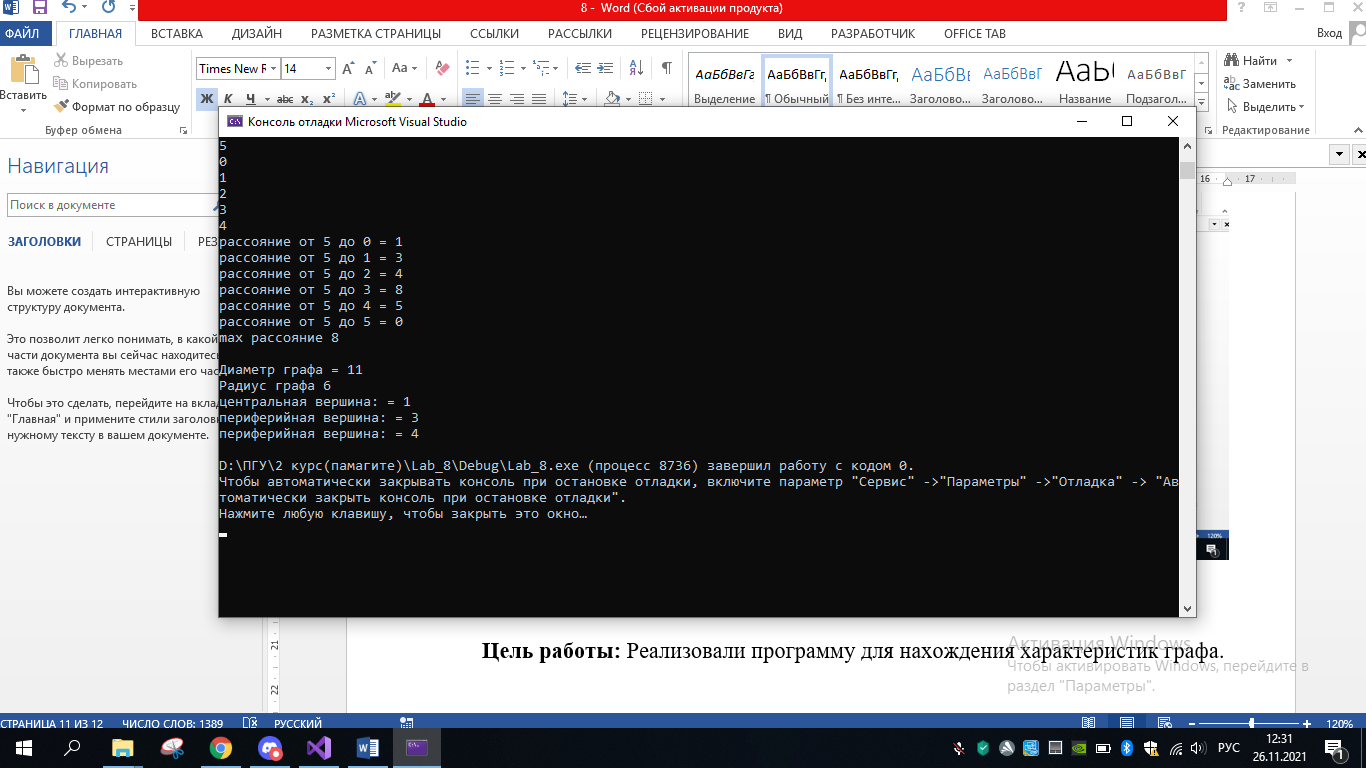












**Цель работы:** Реализовали программу для нахождения характеристик графа.