Министерство образования и науки Российской Федерации

Пензенский государственный университет

Факультет вычислительной техники

**Отчёт**

По лабораторной работе №6

По курсу "ЛиОАвИЗ"

Выполнили:

Студент группы 20ВВ1

Репин И.В.

Приняла:

Юрова О.В.

Пенза 2021

**Название:** Определение характеристик графов

**Цель работы:** выполнить ряд заданий

**Лабораторное задание:**

Задание 1

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.

**Описание метода решения задачи:**

1. Генерируем матрицу смежности для неориентированного взвешенного графа
2. Находим наибольшие среди наименьших расстояний от каждой точки до каждой
3. Находим наибольшее среди предыдущих наибольших
4. Находим наименьшее среди наибольших
5. Перебирая различные точки находим изолированные, доминирующие и т.д.

**Листинг:**

// lb8.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.

//

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <queue>

#include <iostream>

#include <time.h>

using namespace std;

int num, waycntr;

int\*\* m;

int\*\* mi;

int\* visma;

int\* visima;

queue <int> q;

void minDistsMA(int s) {

visma[s] = 0;

q.push(s);

while (!q.empty()) {

int v = q.front();

q.pop();

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

if ((m[v][i] > 0) && (visma[i] > (visma[v] + m[v][i]))) {

q.push(i);

visma[i] = visma[v] + m[v][i];

}

}

}

}

void minDistsIMA(int s) {

visima[s] = 0;

q.push(s);

while (!q.empty()) {

int v = q.front();

q.pop();

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

if (mi[v][i] > 0) {

int l = 0;

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

if ((mi[j][i] > 0)&&(j!=v)) {

l = j;

}

}

if ((visima[l] > (visima[v] + mi[l][i]))&&(mi[l][i] > 0)) {

q.push(l);

visima[l] = visima[v] + mi[l][i];

}

}

}

}

}

int main()

{

printf("Input array size: ");

scanf("%i", &num);

m = new int\* [num];

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

m[i] = new int[num];

}

srand(time(NULL));

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

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

if (j == i) {

m[i][j] = 0;

}

else if (j > i) {

int generate = rand() % 100;

if (generate >= 30) {

m[i][j] = rand() % 100;

}

else {

m[i][j] = 0;

}

}

else {

m[i][j] = m[j][i];

}

}

}

printf("array:");

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

printf("\n");

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

printf("%4i", m[i][j]);

}

}

visma = new int[num];

visima = new int[num];

int\* eccent = new int[num];

int\* centrals = new int[num];

int\* isolated = new int[num];

int\* terminal = new int[num];

int\* dominated = new int[num];

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

visma[i] = 1000;

visima[i] = 1000;

eccent[i] = 0;

centrals[i] = 0;

isolated[i] = 0;

terminal[i] = 0;

dominated[i] = 0;

}

int maxdma = 0, mindma = 1000;

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

minDistsMA(i);

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

if (maxdma < visma[j]) {

maxdma = visma[j];

}

visma[j] = 1000;

}

if ((mindma > maxdma) && (maxdma > 0)) {

mindma = maxdma;

}

}

printf("\nDiameter: %i", maxdma);

printf("\nRadius: %i", mindma);

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

minDistsMA(i);

int max = 0;

bool isolate = true, dominate = true;

int termcntr = 0;

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

if (max < visma[j]) {

max = visma[j];

}

if (m[i][j] > 0) {

isolate = false;

termcntr++;

}

else {

if (i != j) {

dominate = false;

}

}

visma[j] = 1000;

}

if (isolate == true) {

isolated[i] = 1;

}

if (dominate == true) {

dominated[i] = 1;

}

if (termcntr == 1) {

terminal[i] = 1;

}

if (max == maxdma) {

eccent[i] = 1;

}

if (max == mindma) {

centrals[i] = 1;

}

}

printf("\nPeripherals:");

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

if (eccent[i] == 1) {

printf("%2i", i);

}

}

printf("\nCentrals:");

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

if (centrals[i] == 1) {

printf("%2i", i);

}

}

printf("\nIsolated:");

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

if (isolated[i] == 1) {

printf("%2i", i);

}

}

printf("\nTerminal:");

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

if (terminal[i] == 1) {

printf("%2i", i);

}

}

printf("\nDominated:");

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

if (dominated[i] == 1) {

printf("%2i", i);

}

}

waycntr = 0;

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

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

if ((j > i) && (m[i][j]>0)) {

waycntr++;

}

}

}

mi = new int\* [num];

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

mi[i] = new int[waycntr];

}

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

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

mi[i][j] = 0;

}

}

int actway = 0;

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

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

if ((j > i) && (m[i][j] > 0)) {

mi[i][actway] = m[i][j];

mi[j][actway] = m[i][j];

actway++;

}

}

}

printf("\n\nIncident array:");

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

printf("\n");

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

printf("%4i", mi[i][j]);

}

}

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

eccent[i] = 0;

centrals[i] = 0;

isolated[i] = 0;

terminal[i] = 0;

dominated[i] = 0;

}

maxdma = 0, mindma = 1000;

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

minDistsIMA(i);

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

if (maxdma < visima[j]) {

maxdma = visima[j];

}

visima[j] = 1000;

}

if ((mindma > maxdma) && (maxdma > 0)) {

mindma = maxdma;

}

}

printf("\nDiameter: %i", maxdma);

printf("\nRadius: %i", mindma);

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

minDistsIMA(i);

int max = 0;

bool isolate = true, dominate = true;

int termcntr = 0;

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

if (max < visima[j]) {

max = visima[j];

}

if (m[i][j] > 0) {

isolate = false;

termcntr++;

}

else {

if (i != j) {

dominate = false;

}

}

visima[j] = 1000;

}

if (isolate == true) {

isolated[i] = 1;

}

if (dominate == true) {

dominated[i] = 1;

}

if (termcntr == 1) {

terminal[i] = 1;

}

if (max == maxdma) {

eccent[i] = 1;

}

if (max == mindma) {

centrals[i] = 1;

}

}

printf("\nPeripherals:");

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

if (eccent[i] == 1) {

printf("%2i", i);

}

}

printf("\nCentrals:");

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

if (centrals[i] == 1) {

printf("%2i", i);

}

}

printf("\nIsolated:");

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

if (isolated[i] == 1) {

printf("%2i", i);

}

}

printf("\nTerminal:");

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

if (terminal[i] == 1) {

printf("%2i", i);

}

}

printf("\nDominated:");

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

if (dominated[i] == 1) {

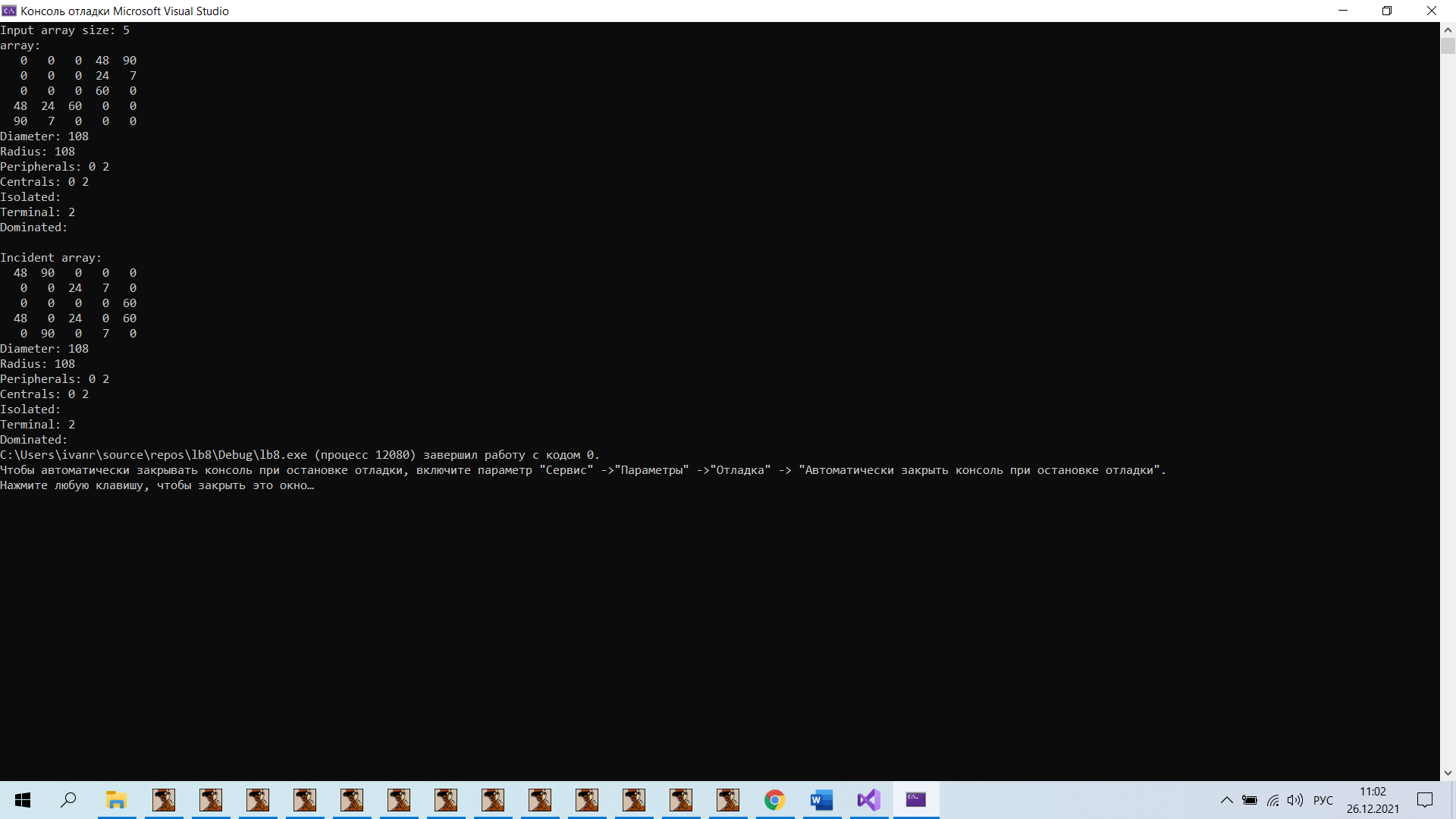
printf("%2i", i);

}

}

}

**Результаты работы программы:**

 **Вывод:** научились определять характеристики графов