Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра "Вычислительная техника"

Отчет

по лабораторной работе №7

по курсу “Л и ОА в ИЗ”

на тему “Обход графа в глубину”

Выполнили студенты группы 22ВВС1:

Агапов И.Е.

Братчиков А.А.

Приняли:

Юрова О.В.

Акифьев И.В.

Пенза 2023

### Лабораторное задание

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.

**Листинг**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

#include <iostream>

#include <stack>

using namespace std;

typedef struct Node {

int value;

struct Node\* next;

} Node;

Node\* create(int data)

{

// Выделение памяти под корень списка

Node\* tmp = (Node\*)malloc(sizeof(Node));

tmp->value = data;

// Присваивание указателю на следующий элемент значения NULL

tmp->next = NULL;

return(tmp);

}

Node\* pushStart(int data, Node\* head)

{

Node\* tmp = create(data);

// Присваивание указателю на следующий элемент значения указателя на «голову»

// первоначального списка

tmp->next = head;

return(tmp);

}

stack <int> st;

bool sum;

Node\*\* sps;

Node\*\* last;

int\*\* mas;

int size;

bool\* Num;

int w;

int y=0;

void DSF(int v,int size) {

Num[v] = true;

printf("%d \t", v + 1);

while(w<size) {

if (mas[v][w] == 1 && Num[w] == false) {

DSF(w,size);

w=0;

}

w++;

}

return;

}

void DSFsps(int v, int size) {

Num[v] = true;

printf("%d \t", v + 1);

while (w < size) {

if (last[v] != NULL) {

if (Num[last[v]->value] == false) {

DSFsps(last[v]->value, size);

}

last[v] = last[v]->next;

}

w++;

}

}

int DSF2(int v,int size) {

Num[v] = true;

st.push(v);

int b= st.top();

while (!st.empty()) {

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

if (mas[b][w] == 1 && Num[w] == false) {

Num[w] = true;

st.push(w);

}

}

b = st.top();

printf("%d \t",b+1);

st.pop();

}

return 0;

}

int main() {

srand(24223);

int size = 0;

sps = (Node\*\*)malloc(sizeof(Node\*) \* size);

last = (Node\*\*)malloc(sizeof(Node\*) \* size);

scanf("%d", &size);

w = size;

Num = (bool\*)malloc(sizeof(bool) \* size);

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

int j = 0;

int i = 0;

setlocale(LC\_ALL, "");

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

Num[k] = false;

}

while (size > i) {

j = 0;

mas[i] = (int\*)malloc(sizeof(int) \* w);

while (w > j) {

mas[i][j] = 0;

j++;

}

i++;

}

i = 0;

j = 0;

while (size > i) {

j = i;

while (w > j) {

if (i == j) {

mas[i][j] = 0;

}

else {

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

mas[j][i] = mas[i][j];

}

j++;

}

i++;

}

i = 0;

j = 0;

bool fl = true;

while (size > i) {

j = 0;

while (w > j) {

if (mas[i][j] == 1) {

if (fl) {

last[i] = pushStart(j, NULL);

fl = false;

}

else {

last[i] = pushStart(j, last[i]);

}

}

j++;

}

i++;

fl = true;

}

printf("\t");

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

printf("%d\t", o + 1);

}

printf("\n");

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

printf("%d\t", o + 1);

for (int n = 0; n < w; n++) {

printf("%d\t", mas[o][n]);

}

printf("\n");

}

printf("\n");

printf("\n");

printf("\n");

w = 0;

scanf("%d", &i);

i--;

/\*DSF2(0, size);\*/

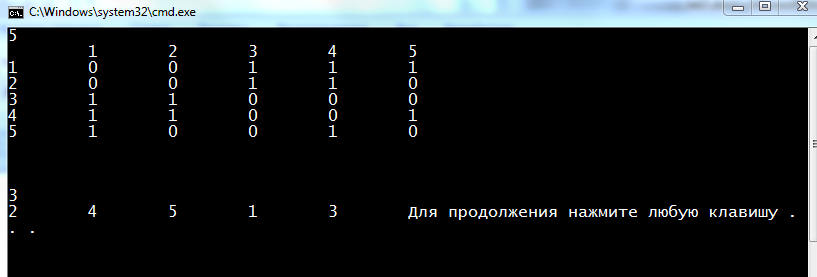
if (Num[i] == false) {

DSF2(i, size);

}

}

**Результат работы программы**

****

**Вывод**

### В ходе выполнения работы были обобщены знания по графам. Были реализованы стандартная функция обхода графа в глубину с рекурсией, функция обхода в глубину для графа, представленного списками смежности.

### Было выполнено преобразование рекурсивной реализации обхода графа к не рекурсивной.