Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №4

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему « Обход графа в глубину»

Выполнил:

студент группы 20ВВ2

Кочергин В.П.

Принял:

к.т.н. доцент Юрова О.В.

д.т.н. профессор Митрохин М. А.

Пенза 2021

**Цель работы:** Реализовать алгоритм обхода графов в глубину.

**Задание 1** Стандартный алгоритм обхода в глубину

Листинг:

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <windows.h>

int num = 0, i, j, \*\* a, \*vis, max, \*parent, ran;

void road(int num, int\*)

{

vis[num] = 1;

for (i = 0; i < max; i++)

{

if (a[num][i] == 1 && vis[i] == 0)

{

parent[i] = num;

printf("%d ", i);

road(i, parent);

}

}

for (i = 0; i < max; i++)

{

if (vis[i] == 0)

{

num = parent[num];

road(num, parent);

}

}

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

setvbuf(stdin, NULL, \_IONBF, 0);

setvbuf(stdout, NULL, \_IONBF, 0);

printf("Введите размеры массива ");

scanf\_s("%d", &max);

printf("Матрица смежности:\n");

a = (int\*\*)malloc(max \* sizeof(int));

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

parent = (int\*)malloc(max \* sizeof(int));

srand(time(NULL));

for (i = 0; i < max; i++)

{

vis[i] = 0;

}

for (i = 0; i < max; i++)

{

a[i] = (int\*)malloc(max \* sizeof(int));

}

for (i = 0; i < max; i++)

{

for (j = 0; j < max; j++)

{

a[i][j] = 0;

}

}

for (i = 0; i < max; i++)

{

for (j = 1 + i; j < max; j++)

{

ran = rand() % 101;

if (ran > 30) a[i][j] = 1;

else a[i][j] = 0;

}

}

for (i = 0; i < max; i++)

{

for (j = 0; j < max; j++)

{

a[j][i] = a[i][j];

}

}

for (i = 0; i < max; i++)

{

for (j = 0; j < max; j++)

{

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

}

printf("\n");

}

printf("\n");

for (i = 0; i < max; i++)

{

parent[i] = 0;

}

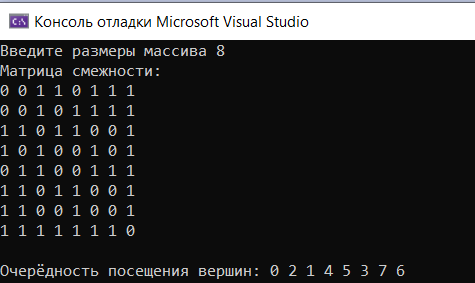
i = 0; j = 0;

printf("Очерёдность посещения вершин: 0 ");

road(num, parent);

}

Результаты работы программы:



**Задание 2** Алгоритм не рекурсивного обхода в глубину.

Листинг:

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <windows.h>

#include <stack>

int num = 0, i, j, \*\* a, \* vis, ran, Max, \* parent;

using namespace std;

void Stack(int)

{

stack <int> q;

q.push(num);

vis[num] = 1;

while (!q.empty())

{

num = q.top();

q.pop();

for (i = 0; i < Max; i++)

{

if (vis[i] == 0 && a[num][i] == 1)

{

q.push(i);

vis[i] = 1;

printf("%d ", i);

num = q.top();

i = 0;

}

}

}

}

void road(int num, int\*)

{

vis[num] = 1;

for (i = 0; i < Max; i++)

{

if (a[num][i] == 1 && vis[i] == 0)

{

parent[i] = num;

printf("%d ", i);

road(i, parent);

}

}

for (i = 0; i < Max; i++)

{

if (vis[i] == 0)

{

num = parent[num];

road(num, parent);

}

}

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

setvbuf(stdin, NULL, \_IONBF, 0);

setvbuf(stdout, NULL, \_IONBF, 0);

printf("Введите размеры массива ");

scanf\_s("%d", &Max);

printf("Матрица смежности:\n");

a = (int\*\*)malloc(Max \* sizeof(int));

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

parent = (int\*)malloc(Max \* sizeof(int));

srand(time(NULL));

for (i = 0; i < Max; i++)

{

vis[i] = 0;

}

for (i = 0; i < Max; i++)

{

a[i] = (int\*)malloc(Max \* sizeof(int));

}

for (i = 0; i < Max; i++)

{

for (j = 0; j < Max; j++)

{

a[i][j] = 0;

}

}

for (i = 0; i < Max; i++)

{

for (j = 1 + i; j < Max; j++)

{

ran = rand() % 101;

if (ran > 30) a[i][j] = 1;

else a[i][j] = 0;

}

}

for (i = 0; i < Max; i++)

{

for (j = 0; j < Max; j++)

{

a[j][i] = a[i][j];

}

}

for (i = 0; i < Max; i++)

{

for (j = 0; j < Max; j++)

{

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

}

printf("\n");

}

printf("\n");

for (i = 0; i < Max; i++)

{

parent[i] = 0;

}

i = 0; j = 0;

printf("Очерёдность посещения вершин с рекурсией: 0 ");

road(num, parent);

for (i = 0; i < Max; i++)

{

vis[i] = 0;

}

num = 0;

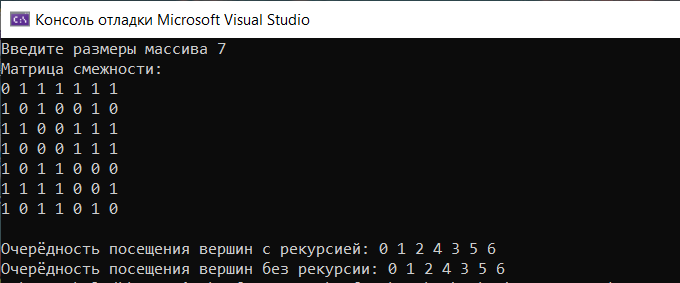
printf("\n");

printf("Очерёдность посещения вершин без рекурсии: 0 ");

Stack(num);

}

Результаты работы программы:



**Задание 3** Алгоритм обхода в глубину на списке смежности

Листинг:

#include <time.h>

#include <malloc.h>

#include <windows.h>

#include <iostream>

using namespace std;

struct Node

{

int items;

Node\* next;

};

typedef Node\* PNode;

PNode\* Head;

void InitializHead(int size)

{

int i;

Head = (PNode\*)malloc(size \* sizeof(PNode));

for (i = 0; i < size; i++) Head[i] = NULL;

}

PNode CreateNode(int items)

{

PNode NewNode = (Node\*)malloc(sizeof(Node));

NewNode->items = items;

NewNode->next = NULL;

return NewNode;

}

void AddFirst(PNode NewNode, int i)

{

NewNode->next = Head[i];

Head[i] = NewNode;

}

void AddAfter(PNode q, PNode NewNode)

{

NewNode->next = q->next;

q->next = NewNode;

}

void AddLast(PNode NewNode, int i)

{

PNode q = Head[i];

while (q->next) q = q->next;

AddAfter(q, NewNode);

}

void print(int i)

{

PNode q = Head[i];

while (q)

{

printf("%d ", q->items);

q = q->next;

}

printf("\n");

}

void road(int size, int\*\* a, int\* vis, int num, int\* parent)

{

vis[num] = 1;

for (int i = 0; i < size; i++)

{

if (a[num][i] == 1 && vis[i] == 0)

{

parent[i] = num;

printf("%d ", i);

road(size, a, vis, i, parent);

}

}

for (int i = 0; i < size; i++)

{

if (vis[i] == 0)

{

num = parent[num];

road(size, a, vis, num, parent);

}

}

}

void road2(int size, int\* vis, int num, int\* parent)

{

vis[num] = 1;

while (Head[num])

{

if (vis[Head[num]->items] == 0)

{

parent[Head[num]->items] = num;

printf("%d ", Head[num]->items);

road2(size, vis, Head[num]->items, parent);

}

Head[num] = Head[num]->next;

}

for (int i = 0; i < size; i++)

{

if (vis[i] == 0)

{

num = parent[num];

road2(size, vis, num, parent);

}

}

}

int main()

{

int size, i, j, ran, \*\* a, \*vis, \*parent, num = 0;

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

PNode NewNode = NULL;

printf("Укажите размер массива: ");

scanf\_s("%d", &size);

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

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

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

for (i = 0; i < size; i++)

{

a[i] = (int\*)malloc(size \* sizeof(int));

}

for (i = 0; i < size; i++)

{

for (j = 0; j < size; j++)

{

a[i][j] = 0;

}

}

for (i = 0; i < size; i++)

{

vis[i] = 0;

}

for (i = 0; i < size; i++)

{

for (j = 1 + i; j < size; j++)

{

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

}

}

for (i = 0; i < size; i++)

{

for (j = 0; j < size; j++)

{

a[j][i] = a[i][j];

}

}

for (i = 0; i < size; i++)

{

for (j = 0; j < size; j++)

{

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

}

printf("\n");

}

printf("Обход в глубину в матрице: 0 ");

road(size, a, vis, num, parent);

for (i = 0; i < size; i++)

{

vis[i] = 0;

}

num = 0;

InitializHead(size);

for (i = 0; i < size; i++)

{

NewNode = CreateNode(i);

AddFirst(NewNode, i);

}

for (i = 0; i < size; i++)

{

for (j = 0; j < size; j++)

{

if (a[i][j] == 1)

{

NewNode = CreateNode(j);

AddLast(NewNode, i);

}

}

}

printf("\n\n");

for (i = 0; i < size; i++)

{

print(i);

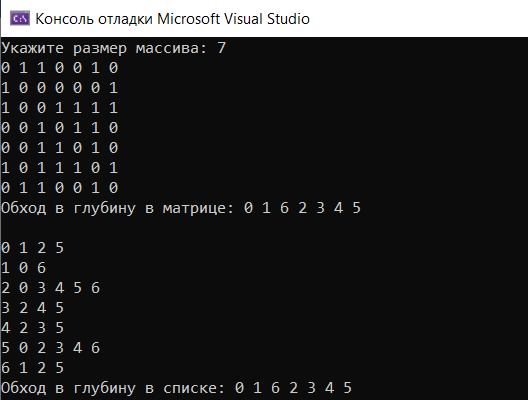
}

printf("Обход в глубину в списке: 0 ");

road2(size, vis, num, parent);

}

Результат работы программы:



Вывод: Сегодня удалось реализовать алгоритм обхода графов в глубину c рекурсией и без неё.