|  |
| --- |
| Министерство образования Российской Федерации  Пензенский государственный университет  Кафедра «Вычислительная техника» |
| Отчет  по лабораторной работе №5  по курсу «Л и ОА в ИЗ»  на тему «Обход графа в ширину» |
|  |
|  |
| Выполнил:  студент группы 20ВВ4  Кривцов Н.А.  Слабинский Н.А.  Приняли:  Юрова О.В.  Акифьев И.В. |
| Пенза 2021 |

# Цель работы: выполнить обход графа в ширину.

**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 ширину,

реализованную в соответствии с приведенным выше описанием. При

реализации алгоритма в качестве очереди используйте класс queue из

стандартной библиотеки С++.

**Теория:**

Обход графа в ширину – еще один распространенный способ обхода

графов.

Основная идея такого обхода состоит в том, чтобы посещать вершины

по уровням удаленности от исходной вершины. Удалённость в данном

случае понимается как количество ребер, по которым необходимо прейти до

достижения вершины.

Класс Queue представляет обычную очередь, работающую по алгоритму FIFO ("первый вошел - первый вышел").

У класса Queue можно отметить следующие методы:

* Dequeue: извлекает и возвращает первый элемент очереди
* Enqueue: добавляет элемент в конец очереди
* Peek: просто возвращает первый элемент из начала очереди без его удаления

**Листинг:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "locale"

#include "stdio.h"

#include "conio.h"

#include <locale.h>

#include <stdlib.h>

#include <malloc.h>

#include <iostream>

int NUM[6] = { 0,0,0,0,0,0 };

void DFS1(int v, int\*\* matrix, int size, int iskl) {

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

if (matrix[v][i] == 1 && NUM[i] == 0)

{

if (iskl != i) {

printf("%d", i);

}

NUM[i] = 1;

}

}

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

if (NUM[i] == 1)

{

NUM[i] = 2;

DFS1(i, matrix, size, iskl);

}

}

}

int main()

{

srand(time(NULL));

setlocale(LC\_ALL, "RUS");

int v, n;

int k = 0;

printf("Введите размер матрицы: ");

scanf("%d", &n);

int\*\* M;

M = (int\*\*)malloc(n \* sizeof(int\*));

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

{

M[i] = (int\*)malloc(n \* sizeof(int));

}

printf("Матрицы:");

printf("\n");

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

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

if (j > k) {

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

printf("%3d", M[i][j]);

}

else if (j < k) {

M[i][j] = M[j][i];

printf("%3d", M[i][j]);

}

else {

M[i][j] = 0;

printf("%3d", M[i][j]);

}

}

k++;

printf("\n");

}

printf("\n\n");

printf("Введите вершину, с которой будет начинаться обход: ");

scanf("%d", &v);

printf("\n");

printf("Результат: ");

printf("%d", v);

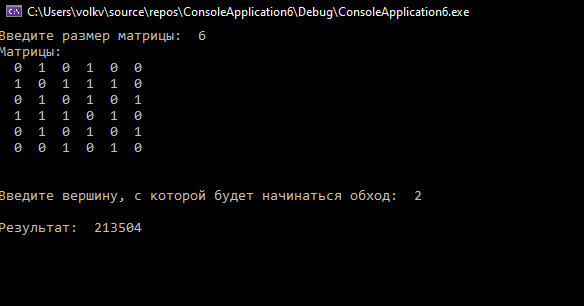
while (NUM[v] != 1)

{

DFS1(v, M, n, v);

}

}



**Вывод:** Разобрали метод обхода графа в ширину и написали код который выполняет это автоматически.