ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  | Рогачев С. А. |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №7 |
| «АЛГОРИТМЫ НА ГРАФАХ» |
| по курсу: Структуры и алгоритмы обработки данных |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | М011 |  |  |  | Борисов С. И. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2021

**Цель работы**

Целью работы является изучение графов и получение практических навыков их использования.

**Задание на лабораторную**

Разработать на языке программирования высокого уровня программу, которая должна выполнять функцию, в соответствии с вариантом задания. Варианты задания приведены в таблице 7 (формулировки задач приведены после таблицы).



Таблица 7.

**Листинг программы, реализующей алгоритм**

// Борисов С.И. ЛБ-7 Вариант 7.

#include <iostream>

using namespace std;

struct Node

{

int inf;

Node\* next;

};

int check(int A) //проверка корректности ввода

{

while (cin.fail() || cin.rdbuf()->in\_avail() > 1 || A < 0) //проверка корректности ввода

{

cout << "Введено недопустимое значение переменной. Пожалуйста, повторите попытку: ";

cin.clear();

cin.ignore((numeric\_limits<streamsize>::max)(), '\n');

cin >> A;

}

return A;

}

void push(Node\*& st, int dat) // Загрузка числа в стек

{

Node\* el = new Node;

el->inf = dat;

el->next = st;

st = el;

}

int pop(Node\*& st) // Извлечение из стека

{

int value = st->inf;

Node\* temp = st;

st = st->next;

delete temp;

return value;

}

int peek(Node\* st)// Получение числа без его извлечения

{

return st->inf;

}

Node\*\* graph; // Массив списков смежности

void add(Node\*& list, int data) //Добавление смежной вершины

{

if (!list) { list = new Node; list->inf = data; list->next = 0; return; }

Node\* temp = list;

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

Node\* elem = new Node;

elem->inf = data;

elem->next = NULL;

temp->next = elem;

}

void del(Node\*& l, int key) // Удаление вершины key из списка

{

if (l->inf == key) { Node\* tmp = l; l = l->next; delete tmp; }

else

{

Node\* tmp = l;

while (tmp)

{

if (tmp->next) // есть следующая вершина

if (tmp->next->inf == key)// и если она искомая

{

Node\* tmp2 = tmp->next;

tmp->next = tmp->next->next;

delete tmp2;

}

tmp = tmp->next;

}

}

}

int eiler(Node\*\* gr, int num) // Определение эйлеровости графа

{

int count;

for (int i = 0; i < num; i++) //проход по всем вершинам

{

count = 0;

Node\* tmp = gr[i];

while (tmp) // считаем степень

{

count++;

tmp = tmp->next;

}

if (count % 2 == 1)return 0; // степень нечетная

}

return 1; // все степени четные

}

void eiler\_path(Node\*\* gr, int vertex)

{ //Построение цикла

Node\* S = NULL; // Стек для пройденных вершин

int v = vertex - 1; // 1я вершина (произвольная)

int u;

push(S, v); //сохраняем ее в стек

while (S)//пока стек не пуст

{

v = peek(S); // текущая вершина

if (!gr[v])// если нет инцидентных ребер

{

v = pop(S); cout << v + 1 << " "; //выводим вершину (у нас отсчет от 1, поэтому +1)

}

else

{

u = gr[v]->inf; push(S, u); //проходим в следующую вершину

del(gr[v], u); del(gr[u], v); //удаляем пройденное ребро

}

}

}

int main()

{

setlocale(LC\_ALL, "Russian");

cout << "Количество вершин: "; int n; cin >> n; // Количество вершин

n = check(n);

int zn; // Текущее значение

int vertex;

graph = new Node \* [n];

for (int i = 0; i < n; i++)graph[i] = NULL;

for (int i = 0; i < n; i++) // заполняем массив списков

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

{

cin >> zn;

if (zn) add(graph[i], j);

}

cout << "Выберети вершину, из которой начнётся поиск Эйлерова пути: ";

do

{

cin >> vertex;

vertex = check(vertex);

} while (vertex > n);

cout << "\n\nРезультат пути: ";

if (eiler(graph, n))eiler\_path(graph, vertex);

else cout << "Граф не является эйлеровым.";

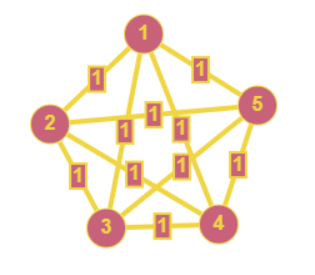
cout << endl;

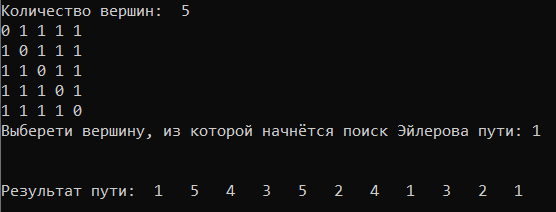
return(0);

}

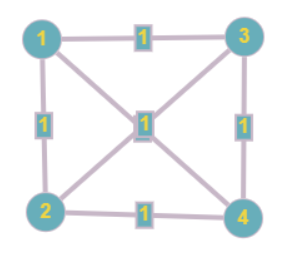
**Тестирование алгоритма:**

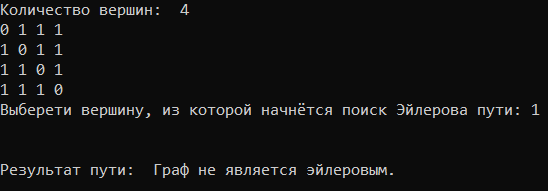
1)





2)





**Выводы**

На основе этих расчетов можно сделать вывод, что был разработан алгоритм, характеристики которого соответствуют поставленному заданию.