ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | М. А. Мурашова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| «АЛГОРИТМЫ НА ГРАФАХ» |
| по курсу: СТРУКТУРЫ И АЛГОРИТМЫ ОБРАБОТКИ ДАННЫХ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4134к |  |  |  | Столяров Н.С. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2022

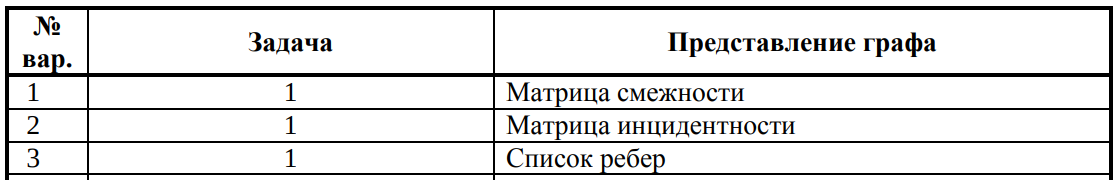
**Цель работы**

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

**Задание на лабораторную работу**

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).

**Вариант 3**



**Листинг**

**main.cpp**

**#include <iostream>**

**using** **namespace** std**;**

#include "libs/lib.h"

#include <cmath>

#include <iomanip>

// проверка ввода

#include "libs/simple\_char.h"

#include "libs/input\_validation.h"

#include "list\_ribs.h"

int main**()** **{**

// смена кодировки

system**(**"chcp 65001"**);**

List\_ribs list\_ribs**(**4**,** 4**);**

list\_ribs**.**link**(**0**,** 0**,** 0**,** 1**);**

list\_ribs**.**link**(**0**,** 1**,** 0**,** 2**);**

list\_ribs**.**link**(**0**,** 2**,** 0**,** 3**);**

list\_ribs**.**link**(**0**,** 3**,** 1**,** 3**);**

list\_ribs**.**link**(**1**,** 3**,** 2**,** 3**);**

list\_ribs**.**link**(**2**,** 3**,** 3**,** 3**);**

list\_ribs**.**link**(**3**,** 3**,** 3**,** 2**);**

list\_ribs**.**link**(**3**,** 2**,** 3**,** 1**);**

list\_ribs**.**link**(**3**,** 1**,** 3**,** 0**);**

list\_ribs**.**link**(**3**,** 0**,** 2**,** 0**);**

list\_ribs**.**link**(**2**,** 0**,** 1**,** 0**);**

list\_ribs**.**link**(**1**,** 0**,** 0**,** 0**);**

list\_ribs**.**link**(**1**,** 1**,** 1**,** 2**);**

list\_ribs**.**link**(**2**,** 1**,** 2**,** 2**);**

list\_ribs**.**show\_list**();**

**if** **(**list\_ribs**.**check**())** **{**

cout **<<** "Граф соответсвует изображению (нет пропущенных пикселей)" **<<** endl**;**

**}** **else** **{**

cout **<<** "Граф не соответсвует изображению (есть пропущенные пиксели)" **<<** endl**;**

**}**

list\_ribs**.**show\_image**();**

**return** 0**;**

**}**

**list\_ribs.h**

#include <iostream>

**using** **namespace** std**;**

#include <vector>

#include <iomanip>

#define TABLE\_ELEM\_SIZE 4

class List\_ribs **{**

public**:**

List\_ribs**(**int**,** int**);**

bool link**(**int**,** int**,** int**,** int**);**

bool check**();**

bool check\_link**(**int**\*);**

void show\_list**();**

void show\_image**();**

bool check\_loop**(**int**\*);**

vector**<**int**\*>** fill**();**

private**:**

vector**<**int**\*>** links**;**

int width**;**

int height**;**

**};**

List\_ribs**::**List\_ribs**(**int Width**,** int Height**)** **{**

width **=** Width**;**

height **=** Height**;**

**}**

bool List\_ribs**::**link**(**int link\_xu**,** int link\_yu**,** int link\_xw**,** int link\_yw**)** **{**

**if** **(((**link\_xu **>=** width**)** **||** **(**link\_yu **>=** height**)** **||** **(**link\_xw **>=** width**)** **||** **(**link\_yw **>=** height**))** **||** **((**link\_xu **==** link\_xw**)** **&&** **(**link\_yu **==** link\_yw**)))**

**return** **false;**

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

**if** **(**links**.**at**(**i**)[**0**]** **==** link\_xu **&&** links**.**at**(**i**)[**1**]** **==** link\_yu **&&** links**.**at**(**i**)[**2**]** **==** link\_xw **&&** links**.**at**(**i**)[**3**]** **==** link\_yw**)**

**return** **false;**

**}**

int**\*** arr **=** **new** int**[**4**];**

arr**[**0**]** **=** link\_xu**;**

arr**[**1**]** **=** link\_yu**;**

arr**[**2**]** **=** link\_xw**;**

arr[3] = link\_yw;

links.push\_back(arr);

return true;

}

bool List\_ribs::check\_link(int\* link) {

bool ok;

for (int x = 0; x < width; x++) {

for (int y = 0; y < height; y++) {

if ((link[0] == x && link[1] == y) || (link[2] == x && link[3] == y)) {

return true;

}

}

}

return false;

}

bool List\_ribs::check() {

bool ok;

for (int x = 0; x < width; x++) {

for (int y = 0; y < height; y++) {

ok = false;

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

if ((links.at(i)[0] == x && links.at(i)[1] == y) || (links.at(i)[2] == x && links.at(i)[3] == y)) {

ok = true;

}

}

if (!ok) return false;

}

}

return true;

}

void List\_ribs::show\_list() {

cout << "|" << setw(TABLE\_ELEM\_SIZE \* 2) << "U" << " |" << setw(TABLE\_ELEM\_SIZE \* 2) << "W" << " |" << endl;

cout << "|" << setw(TABLE\_ELEM\_SIZE) << "x" << "|" << setw(TABLE\_ELEM\_SIZE) << "y" << "|";

cout << setw(TABLE\_ELEM\_SIZE) << "x" << "|" << setw(TABLE\_ELEM\_SIZE) << "y" << "|" << endl;

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

cout << "|" << setw(TABLE\_ELEM\_SIZE) << links.at(i)[0];

cout << "|" << setw(TABLE\_ELEM\_SIZE) << links.at(i)[1];

cout << "|" << setw(TABLE\_ELEM\_SIZE) << links.at(i)[2];

cout << "|" << setw(TABLE\_ELEM\_SIZE) << links.at(i)[3] << "|" << endl;

}

}

bool List\_ribs::check\_loop(int\* list) {

int\* current\_list = list;

bool ok;

while (true) {

ok = false;

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

if ((list[0] == current\_list[2]) && (list[1] == current\_list[3])) return true;

if ((current\_list[2] == links.at(i)[0]) && (current\_list[3] == links.at(i)[1])) {

current\_list = links.at(i);

ok = true;

break;

}

}

if (!ok) return false;

}

}

vector<int\*> List\_ribs::fill() {

vector<int\*> fill\_list;

bool ok;

int\* current\_list;

for (int l = 0; l < links.size(); l++) {

if (check\_loop(links.at(l))) {

current\_list = links.at(l);

fill\_list.push\_back(current\_list);

ok = true;

while (ok) {

ok = false;

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

if ((links.at(l)[0] == current\_list[2]) && (links.at(l)[1] == current\_list[3])) ok = false;

if ((current\_list[2] == links.at(i)[0]) && (current\_list[3] == links.at(i)[1])) {

current\_list = links.at(i);

fill\_list.push\_back(current\_list);

break;

}

}

}

}

}

return fill\_list;

}

void List\_ribs::show\_image() {

int\*\* arr = new int\*[width];

int i, j, x, y;

for (i = 0; i < width; i++) {

arr[i] = new int[height];

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

arr[i][j] = 0;

}

for (i = 0; i < links.size(); i++) {

arr[links.at(i)[0]][links.at(i)[1]] = 1;

arr[links.at(i)[2]][links.at(i)[3]] = 1;

}

// cout << check\_loop(links.at(0)) << endl;

vector<int\*> fill\_list = fill();

for (i = 0; i < fill\_list.size(); i++) {

arr[fill\_list.at(i)[0]][fill\_list.at(i)[1]] = 2;

}

for (x = 0; x < width; x++) {

for (y = 0; y < height; y++) {

if (arr[x][y] == 0)

cout << "░░";

if (arr[x][y] == 1)

cout << "▒▒";

if (arr[x][y] == 2)

cout << "▓▓";

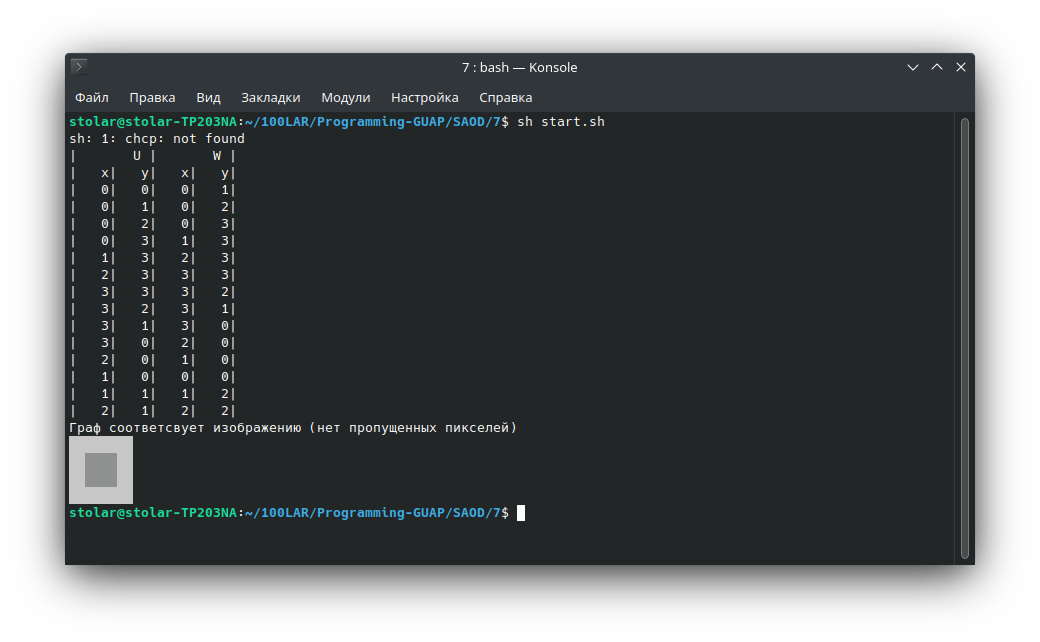
}

cout << endl;

}

}

**Результат работы**

****

**Вывод**

Мы изучили графы и получили практические навыки их использования.