Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут ім. Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації та управління

ЗВІТ

про виконання лабораторного практикуму №7(2)

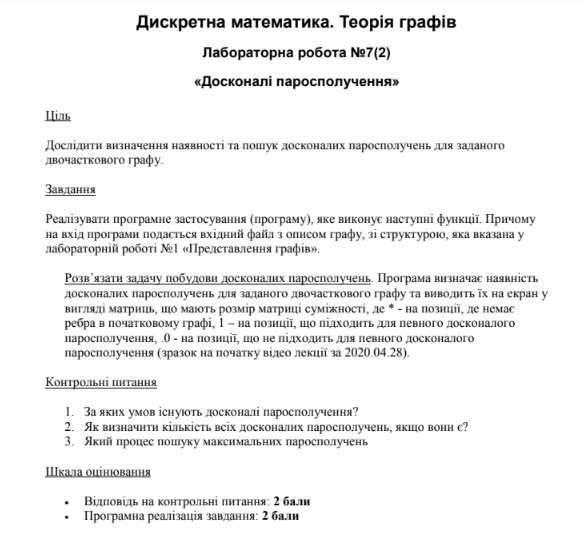
Виконав:

студент 1- го курсу ФІОТ

групи *ІП-91*

*Кінчур Вадим Вікторович*

Київ 2020

1. Умова лабораторної роботи
2. Програмний код (С++):

#include <iostream>

#include <fstream>

#include <list>

#include <vector>

#include <queue>

#include <iomanip>

using namespace std;

struct two\_parts {

vector<int> first, second;

};

void get\_input(const char\*, list<int>\*&, int&);

vector<int> get\_bipartition(list<int>\*, int);

two\_parts get\_two\_parts(vector<int>);

vector<int\*> get\_matches(list<int>\*, vector<int>);

void find\_matches(vector<int\*>&, list<int>\*, vector<int>&, int&, int);

char\*\* get\_template\_matrix(list<int>\*, two\_parts);

void print\_match(int\*, vector<int>, vector<int>, char\*\*);

void print\_matrix(char\*\*, int, int, vector<int>);

void print\_matches(list<int>\*,int);

void print\_vector(vector<int>);

int main() {

list<int>\* adj\_list = nullptr;

int v\_num;

get\_input("input.txt", adj\_list, v\_num);

print\_matches(adj\_list, v\_num);

}

void get\_input(const char\* file\_name, list<int>\*& adj\_l, int& v\_num) {

ifstream file\_handler(file\_name);

int e\_num, start, end;

file\_handler >> v\_num >> e\_num;

adj\_l = new list<int>[v\_num];

for (auto i = 0; i < e\_num; ++i) {

file\_handler >> start >> end;

adj\_l[start - 1].push\_back(end - 1);

adj\_l[end - 1].push\_back(start - 1);

}

file\_handler.close();

}

vector<int> get\_bipartition(list<int>\* adj\_l, int v\_num) {

vector<int> bipart(v\_num, -1);

queue<int> que;

for (auto i = 0; i < v\_num; ++i) {

if (bipart[i] != -1 || adj\_l[i].empty()) continue;

que.push(i); bipart[i] = 0;

while (!que.empty()) {

int start = que.front();

que.pop();

for (auto v : adj\_l[start]) {

if (bipart[v] == -1) {

bipart[v] = 1 - bipart[start];

que.push(v);

}

else if (bipart[v] == bipart[start]) {

return vector<int>();

}

}

}

}

return bipart;

}

two\_parts get\_two\_parts(vector<int> bipart) {

two\_parts res;

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

if (bipart[i]) res.first.push\_back(i);

else res.second.push\_back(i);

}

if (res.second.size() < res.first.size())

swap(res.second, res.first);

return res;

}

void find\_matches(vector<int\*>& matching, list<int>\* adj\_l, vector<int>& l\_part, int& to, int at) {

for (auto& v : adj\_l[l\_part[at]]) {

auto finded = false;

for (auto i = 0; i < at; ++i) {

if (matching[to][i] == v) {

finded = true; break;

}

}

if (finded) continue;

matching[to][at] = v;

if (at == l\_part.size() - 1) {

matching.push\_back(new int[l\_part.size()]);

for (auto i = 0; i < l\_part.size() - 1; ++i) {

matching[to + 1][i] = matching[to][i];

}

++to; return;

}

find\_matches(matching, adj\_l, l\_part, to, at + 1);

}

if (!at) matching.pop\_back();

}

vector<int\*> get\_matches(list<int>\* adj\_l, vector<int> l\_part) {

vector<int\*> matching{ new int[l\_part.size()] };

int to = 0;

find\_matches(matching, adj\_l, l\_part, to, 0);

return matching;

}

char\*\* get\_template\_matrix(list<int>\* adj\_l, two\_parts parts) {

auto\*\* t\_matrix = new char\*[parts.first.size()];

for (auto i = 0; i < parts.first.size(); ++i) {

t\_matrix[i] = new char[parts.second.size()];

}

for (auto i = 0; i < parts.first.size(); ++i) {

for (auto j = 0; j < parts.second.size(); ++j) {

auto el = find(adj\_l[parts.first[i]].begin(),

adj\_l[parts.first[i]].end(), parts.second[j]);

t\_matrix[i][j] = (el != adj\_l[parts.first[i]].end()) ? '0' : '\*';

}

}

return t\_matrix;

}

void print\_matrix(char\*\* matrix, int row\_num, int col\_num, vector<int> l\_part) {

for (auto i = 0; i < row\_num; ++i) {

cout << setw(4) << l\_part[i] + 1;

for (auto j = 0; j < col\_num; ++j) {

cout << setw(4) << matrix[i][j];

matrix[i][j] = (matrix[i][j] != '\*') ? '0' : '\*';

}

cout << endl;

}

}

void print\_vector(vector<int> vec) {

cout << " ";

for (auto el : vec) {

cout << setw(4) << el + 1;

}

cout << endl;

}

void print\_match(int\* match, vector<int> l\_part, vector<int> r\_part, char\*\* t\_matrix) {

print\_vector(r\_part);

for (auto i = 0; i < l\_part.size(); ++i) {

auto index = distance(r\_part.begin(), find(r\_part.begin(), r\_part.end(), match[i]));

t\_matrix[i][index] = '1';

}

print\_matrix(t\_matrix, l\_part.size(), r\_part.size(), l\_part);

}

void print\_matches(list<int>\* adj\_l, int v\_num) {

auto bipart = get\_bipartition(adj\_l, v\_num);

if (bipart.size() == 0) {

cout << "Graph is not biparted\n";

return;

}

auto parts = get\_two\_parts(bipart);

auto\*\* t\_matrix = get\_template\_matrix(adj\_l, parts);

auto matching = get\_matches(adj\_l, parts.first);

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

cout << "Matching #" << i + 1 << ':' << endl;

print\_match(matching[i], parts.first, parts.second, t\_matrix);

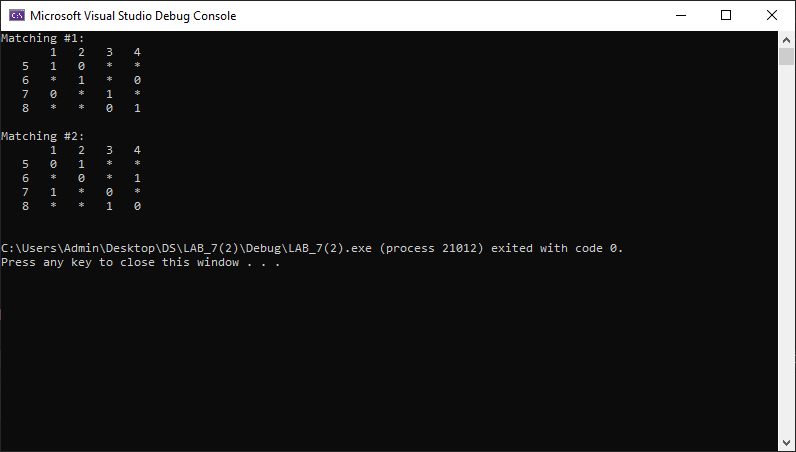
cout << endl;

}

}

1. Результат виконання:

**Вхідні дані:**

8 8

1 5

1 7

2 5

2 6

3 7

3 8

4 6

4 8