Національний технічний університет України

«Київський політехнічний інститут»

Кафедра автоматизованих систем обробки інформації та управління

Звіт про виконання лабораторної роботи №9 з дисципліни

«Алгоритми та структури даних»

Перевірено: Ковалюк Т.В. Виконав ст. 1 курсу ФОІТ

Проскура С.Л. гр. ІС-52

Дорошенко А.В.

Київ 2016

## Лабораторна робота 9

# **Алгоритми на графах. Потоки та паросполучення**

Варіант № 9

*Мета:*

1. Вивчити особливості використання  структур типу графів.

2. Навчитися застосовувати структури типу графів у практичних задачах.

3. Опанувати алгоритми:

* Алгоритм пошукуланцюга, що збільшує потік
* Форда-Фалкерсона

### Завдання

|  |  |
| --- | --- |
| Вася і Сережа грають в наступну гру. У деяких клітинах картатого листка Сережа малює один з символів 'H ', 'O ', 'N' або 'C ', після чого Вася повинен провести між деякими символами, що знаходяться в сусідніх клітинах, лінії так, щоб вийшло коректне зображення хімічної молекули. На жаль, Сережа любить малювати багато символів, і Вася не може відразу визначити, чи можливо взагалі намалювати лінії потрібним способом. Допоможіть йому написати програму, яка дасть відповідь на це питання. У цьому завданні проведені між символами хімічних елементів лінії вважатимемо коректним зображенням молекули, якщо вони задовольняють наступним умовам:  – кожна лінія сполучає символи, намальовані в сусідніх (по стороні) клітинах,  – між кожною парою символів проведено не більш за одну лінію, – від кожного елементу відходить рівно стільки ліній, яка валентність цього елементу (1 для H, 2 для O, 3 для N, 4 для C), – порожні клітини ні з чим не сполучені, – хоч би в одній клітині намальований якийсь символ.  **Вхідні дані:**Перший рядок вхідного файлу містить два натуральні числа n і m (1<=n;m<=50 ) - розміри листочка, на якому малює Сережа. Далі слідують n рядків по m символів в кожному, що задають конфігурацію хімічних елементів, яку намалював Сережа; порожні клітини задаються символом '.'.  **Вихідні результати:**У виихідний файл виведіть одне слово "Valid", якщо лінії провести необхідним чином можна, і "Invalid", якщо не можна/   *Приклад введення:* 3 4  HOH.  NCOH  OO   *Приклад виведення:* Valid |  |

### Блок-схема алгоритму

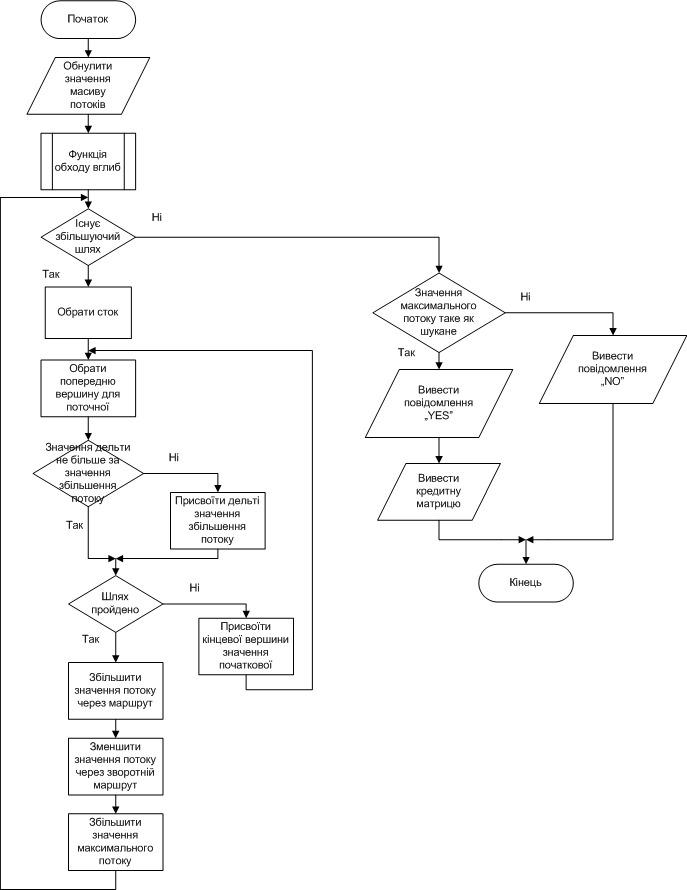
****

Рис.1 (Блок-схема алгоритму знаходження максимального потоку)

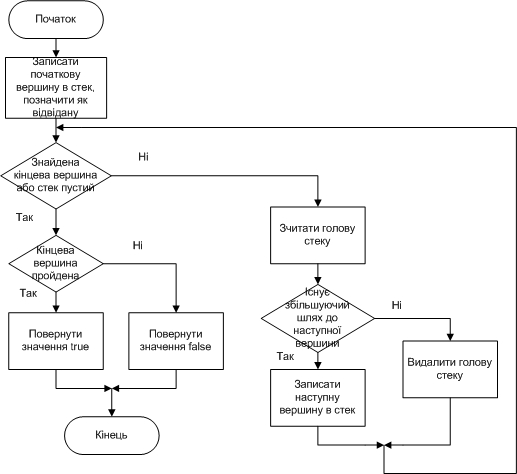
****

Рис.2 (Блок-схема алгоритму обходу вглиб)



Рис.3 (Блок-схема головного алгоритму)

### Код програми

//Laboratory work

//written by student of the first curse

//of the group IC-52

//Anton Doroshenko

//2016.05.04

//==========================================================================

#include <iostream>

#include <cstdio>

#include <algorithm>

#include <vector>

#include <cmath>

#include <set>

#include <map>

#include <string>

#include <cstring>

using namespace std;

const int oo = 1 << 30;

const int neftemu = -(1000000100);

int a, h, n, m;

vector <int> f[2505], g[2505], c[2505], rev[2505], ok[2505];

bool used[2505];

//пошук в глиб

int dfs(int v, int C, int d) {

used[v] = true;

if (v == n - 1)

return C;

for (int i = 0; i < g[v].size(); i++)

if (!used[g[v][i]] && c[v][i] - f[v][i] >= d) {

int k = dfs(g[v][i], min(C, c[v][i] - f[v][i]), d);

if (k) {

f[v][i] += k, f[g[v][i]][rev[v][i]] -= k;

return k;

}

}

return 0;

}

vector <pair <int, int> > e;

int t[2505][2505];

int ok1(char el) {

if (el == 'H')

return 1;

if (el == 'O')

return 2;

if (el == 'N')

return 3;

if (el == 'C')

return 4;

return 0;

}

//Дооавання ребра

void add\_edge(int u, int v, int w) {

e.push\_back(make\_pair(u, v));

f[u].push\_back(0);

c[u].push\_back(w);

g[u].push\_back(v);

ok[u].push\_back(true);

rev[u].push\_back(g[v].size());

f[v].push\_back(0);

c[v].push\_back(0);

ok[v].push\_back(false);

g[v].push\_back(u);

rev[v].push\_back(g[u].size() - 1);

t[u][v] = e.size() - 1;

}

int num[51][51];

//Головна функція

int main() {

cout << "laboratory work number 8 made by Anton Doroshenko, IS-52" << endl;

cout << endl;

cout << "Enter size of sheet" << endl;

int h, w;

cin >> h >> w;

cout << "Enter elements" << endl;

n = 1;

string st[8888];

int cnt = 1;

bool t = false;

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

cin >> st[i];

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

num[i][j] = i \* w + j + 1;

if (st[i][j] != '.')

t = true;

}

}

if (!t) {

puts("Invalid");

return 0;

}

int potok = 0;

n = h\*w + 2;

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

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

if (((i & 1) && (j & 1)) || (!(i & 1) && !(j & 1))) {

add\_edge(0, num[i][j], ok1(st[i][j]));

potok += ok1(st[i][j]);

if (i + 1 < h)

add\_edge(num[i][j], num[i + 1][j], 1);

if (j + 1 < w)

add\_edge(num[i][j], num[i][j + 1], 1);

if (i - 1 >= 0)

add\_edge(num[i][j], num[i - 1][j], 1);

if (j - 1 >= 0)

add\_edge(num[i][j], num[i][j - 1], 1);

}

else

add\_edge(num[i][j], n - 1, ok1(st[i][j]));

}

long long s = 0ll;

for (int i = (1 << 30); i >= 1; i >>= 1) {

while (1) {

memset(used, false, sizeof used);

int k = dfs(0, oo, i);

if (!k)

break;

s += k;

}

}

if (s == potok)

cout << "Valid";

else

cout << "Invalid";

return 0;

}

1. **Screen Shot результатів**

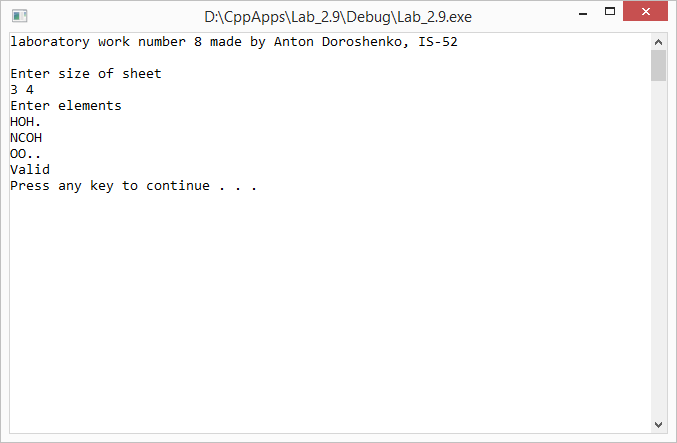


Рис.9 (Screen Shot результатів 1)

1. **Аналіз результатів**

Як ми бачимо зі ScreenShot’ів

Програма приймає розміри листка та послідовність хімічних елементів і виводить повідомлення чи можливо з’єднати елементи так щоб вийшла молекула.

1. **Висновок**

Програма працює правильно, про що свідчить аналіз результатів та ScreenShot’и. Програма коректно виводить результати. Програма застосовує алгоритм Форда - Фалкерсона.