**РУСЕНСКИ УНИВЕРСИТЕТ “АНГЕЛ КЪНЧЕВ”**

**КУРСОВА РАБОТА**

**по Изкуствен интелект**

Студент:

Факултетен номер:

Група:

Специалност: **КСТ**

Курс:

**Изготвил:**

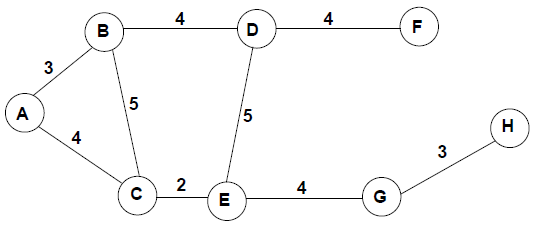
**Дата: Проверил:**

# Задание.

**“Търсене на път по карта” (Map traversal problem):**

Да се намери път между два града, използвайки пътната карта на *Фигура 1*

Алгоритъм: търсене в дълбочина [*Depth-first search (DFS)*].



Фигура 1 Пътна карта

1. **Представяне на задачата като задача за търсене в ПС.**

Т.к могат да се изберат кои да са два върха, нека A е началният връх, а H е крайният. Нека разгледаме постъпковото изпълнения на алгоритъма:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **X** | **Наследници на Х** | **"Оцелели" наследници на Х** | **OPEN** | **CLOSED** | **Коментар** |
| 0 | - | - | - | [A] | [] | Инициализация |
| 1 | A | [B,C] | [B,C] | [B,C] | [A] |  |
| 2 | B | [A,C,D] | [D] | [D,C] | [A,B] | A – CLOSED; C – OPEN |
| 3 | D | [B,E,F] | [E,F] | [E,F,C] | [A,B,D] | B – CLOSED; |
| 4 | E | [C,D,G] | [G] | [G,F,C] | [A,B,D,E] | C – OPEN; D – CLOSED |
| 5 | G | [E,H] | [H] | [H,F,C] | [A,B,D,E,G] | E – CLOSED |
| 6 | H |  |  |  |  | **SUCCESS** |

# Програмна реализация.

## Псевдо-код на алгоритъма.

path[n], count;

procedure **dfs(START, GOAL)**;

begin

if (START = GOAL) begin

print(path);

return;

end

маркиране на посетения връх START;

path[count++] = START; {върхът START участва в пътя}

рекурсия за всички съседи на START, които не са маркирани;

размаркиране на посетения връх START

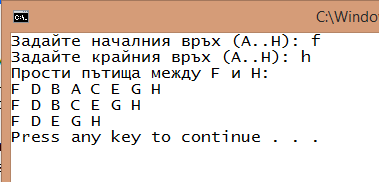
count--;

end; {dfs}

## Описание на разработениете модули.

* **Вид на функцията**: void allDFS(unsigned i, unsigned j)
* **Действие**: Функцията служи за намиране на всички прости пътища между върховете i и j, чрез *Depth-first search* алгоритъма;
* **Параметри**: unsigned i, j – номерата съответно на началния и крайния връх;
* **Използвани функции**: рекурсия
* **Вид на функцията**: int main(void)
* **Действие**: Главна функция
* **Параметри**: няма;
* **Използвани функции**: DFS().

## Тестови примери.



Програмата намира всички прости пътища между два върха в един граф, а за да се изведе само първият намерен път е небходимо след връщане от рекурсия да не се размаркира посетеният връх (и да не се изпълнява count--).

## Код на програмата.

#include <stdio.h>

#include <iostream>

// Брой върхове в графа

const unsigned n = 8;

// Матрица на съседство на графа

const char A[n][n] = {

{ 0, 1, 1, 0, 0, 0, 0, 0 }, // A

{ 1, 0, 1, 1, 0, 0, 0, 0 }, // B

{ 1, 1, 0, 0, 1, 0, 0, 0 }, // C

{ 0, 1, 0, 0, 1, 1, 0, 0 }, // D

{ 0, 0, 1, 1, 0, 0, 1, 0 }, // E

{ 0, 0, 0, 1, 0, 0, 0, 0 }, // F

{ 0, 0, 0, 0, 1, 0, 0, 1 }, // G

{ 0, 0, 0, 0, 0, 0, 1, 0 }, // H

};

char used[n];

unsigned path[n], count;

// Намира всички прости пътища между върховете i и j

void allDFS(unsigned i, unsigned j)

{

unsigned k;

if (i == j) {

path[count] = j;

for (unsigned ind = 0; ind <= count; ind++)

printf("%c ", path[ind] + 'A');

printf("\n");

return;

}

// маркиране на посетения връх

used[i] = 1;

path[count++] = i;

for (k = 0; k < n; k++) // рекурсия за всички съседи на i

if (A[i][k] && !used[k])

allDFS(k, j);

// връщане: размаркиране на посетения връх

used[i] = 0;

count--;

}

int main(void) {

setlocale(LC\_ALL, "BGR");

char sv; // Начален връх

char ev; // Краен връх

for (unsigned k = 0; k < n; k++)

used[k] = 0;

count = 0;

printf("Задайте началния връх (A..%c): ", n + 64);

scanf("%c", &sv);

sv = toupper(sv);

std::cin.sync();

printf("Задайте крайния връх (A..%c): ", n + 64);

scanf("%c", &ev);

ev = toupper(ev);

printf("Прости пътища между %c и %c: \n", sv, ev);

allDFS(sv - 'A', ev - 'A');

return 0;

}