

# Slovenská technická univerzita

Fakulta informatiky a informačných technológií

Ilkovičova 3, 842 19 Bratislava 4

---

**Umelá inteligencia**

**Prehľadávanie stavového priestoru**

**Bláznivá križovatka**

---

Vypracoval: Jozef Varga

Cvičiaci: Mgr. Irina Malkin Ondik, PhD.

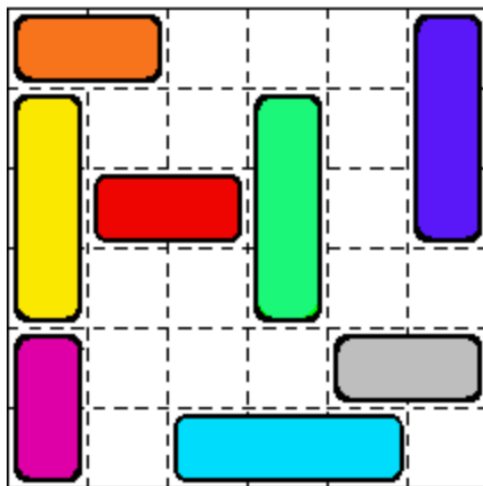
Ak. rok: 2017/2018

## Zadanie

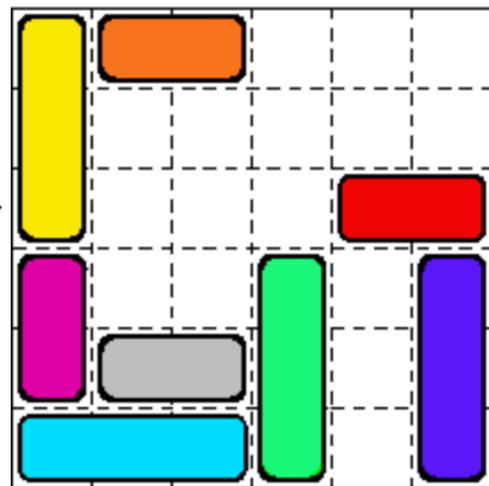
Úlohou je nájsť riešenie hlavolamu Bláznivá križovatka. Hlavolam je reprezentovaný mriežkou, ktorá má rozmery 6 krát 6 políček a obsahuje niekoľko vozidiel (áut a nákladiakov) rozložených na mriežke tak, aby sa neprekrývali. Všetky vozidlá majú šírku 1 políčko, autá sú dlhé 2 a nákladiaky sú dlhé 3 políčka. V prípade, že vozidlo nie je blokované iným vozidlom alebo okrajom mriežky, môže sa posúvať dopredu alebo dozadu, nie však do strany, ani sa nemôže otáčať. V jednom kroku sa môže pohybovať len jedno vozidlo. V prípade, že je pred (za) vozidlom voľných  $n$  políček, môže sa vozidlo pohnúť o 1 až  $n$  políček dopredu (dozadu). Ak sú napríklad pred vozidlom voľné 3 políčka (napr. oranžové vozidlo na počiatočnej pozícii, obr. 1), to sa môže posunúť buď o 1, 2, alebo 3 políčka.

Použite algoritmus prehľadávania do šírky a do hĺbky. Porovnajte ich výsledky.

Počiatočná pozícia



Cieľová pozícia



Obrázok 1 Počiatočná a cieľová pozícia hlavolamu Bláznivá križovatka

## Opis riešenia a použitý algoritmus

Na vypracovanie tohoto zadanie som využil prehľadávanie do hĺbky a do šírky. Ako prvé som riešil prehľadávanie do šírky. Na prehľadávanie do šírky som využil rad (queue). Teda údaje ktoré prvé zapíšem, následne aj ako prvé prečítam a spracujem (FIFO). Keď som riešil prehľadávanie do hĺbky, tak jedinou zmenou bola výmena využívaného radu (queue) za zásobník (stack). Zásobník funguje ako LIFO, čo posledné vložím, to ako prvé vyťahujem. Môj algoritmus funguje nasledovne:

1. Užívateľ zadá potrebné údaje o autách (prvý stav)
2. Stav rozpoloženia áut vložím do radu/zásobníka a hash mapy
3. Vyberie z radu/ zásobníka stav (ak je pri vyberaní prázdny, riešenie neexistuje algoritmus končí)
4. Vyberiem auto ktoré ešte nebolo skontrolované v danom stave, ak také neexistuje vrátim sa na krok 3.
5. Prechádza auto a kontrolujem či sa môže posunúť teda či v smere pohybu nie je iné auto alebo či by sme pohybom auta nevyčnievali z mapy. Tu sú 2 možnosti:
  - a. Auto je horizontálne otočené teda riešim či sa je s autom možné pohnúť doľava alebo doprava
  - b. Auto je otočené vertikálne a tým pádom riešim posun hore alebo dole
6. Ak sa je možné pohnúť, pohnem s autom a skontrolujem v hash mape či v nej takýto stav už náhodou neexistuje, ak taký stav neexistuje vložím ho do radu/ zásobníka a hash mapy, pokračujem na krok 7. Do hash mapy vkladám predošlú a aktuálnu mapu. Ak nie je možné pohnúť autom, alebo takýto stav už v hash mape existuje, skočím na krok 4.
7. Skontrolujem či náš nový stav nie je náhodou výsledok („červené“ autíčko napravo). Ak je skončím prehľadávanie a pomocou hash tabuľky spätne vypíšem postupnosť krokov ináč idem na krok 4.

# Reprezentácia údajov

## Stav

Vstupom algoritmov je začiatkový stav, ten je zadávaný v tvare stringu. Jedno auto reprezentujú štyri znaky stringu v tvare „XYDO“

X -> x-ová súradnica

Y -> y-ová súradnica

D -> dĺžka auta (v našom prípade 2 alebo 3)

O -> orientácia autíčka

    či je vertikálne natočené : 0

    alebo horizontálne natočené: 1

Stav vyzerá nasledovne :

„12210021503004203130013044212531“

Pričom každé štyri znaky sú iné auto. Prvé auto je podľa obr.1 „červené“, teda cieľový stav je definovný že prvé auto je na najpravejšej pozícii v riadku. To vo všeobecnosti definuje celú množinu cieľových stavov a nás nezaujímá, ktorý z nich bude vo výslednom riešení.

Každému auto sa pri výpise prideli písmeno v abecede. Prvé (to ktoré sa má dostať do cieľa) je reprezentované písmenom A (1221), ďalšie B (0021), ďalšie C ...

## Operátory

Sú len štyri a to VPRAVO, VLAVO, HORE, DOLE

Tvar (VPRAVO písmeno\_vozidla), (VLAVO písmeno\_vozidla), (HORE písmeno\_vozidla), (DOLE písmeno\_vozidla)

Príklad použitia operátora (VPRAVO B):

Vstupný stav:

„12210021503004203130013044212531“

Výstupný stav:

„12211021503004203130013044212531“

## Uzol

Uzol je reprezentovaný ako hash mapa ktorá nám uchováva dvojice aktuálny stav (kľúč v hash mape) a predchádzajúci stav (hodnota v hash mape)

## Užívateľské prostredie

Program užívateľa požiada o zadanie počtu áut a následne o zadanie ich údajov v tvare:

„XYDO“

X -> x-ová súradnica

Y -> y-ová súradnica

D -> dĺžka auta (v našom prípade 2 alebo 3)

O -> orientácia autíčka

    či je vertikálne natočené : 0

    alebo horizontálne natočené: 1

```
Pomocka:
X = x-ova suradnica
Y = y-ova suradnica
D = dlzka (2/3)
O = orientacia (0 - vertikálne/ 1 - horizontálne)
Auticka budu pridelené písmena (1. auticko = A, 2. auticko B, ...)
Auticka zadavajte v tvare 'XYDO' enter
Ako prve pri zadavani auticok zadajte unikove vozidlo.

Zadajte pocet aut:
8
Zadajte auticka:
1221
0021
5030
0420
3130
0130
4421
2531
```

Obrázok 2 spustenie programu a ukážka vstupu

(program predpokladá korektný vstup)

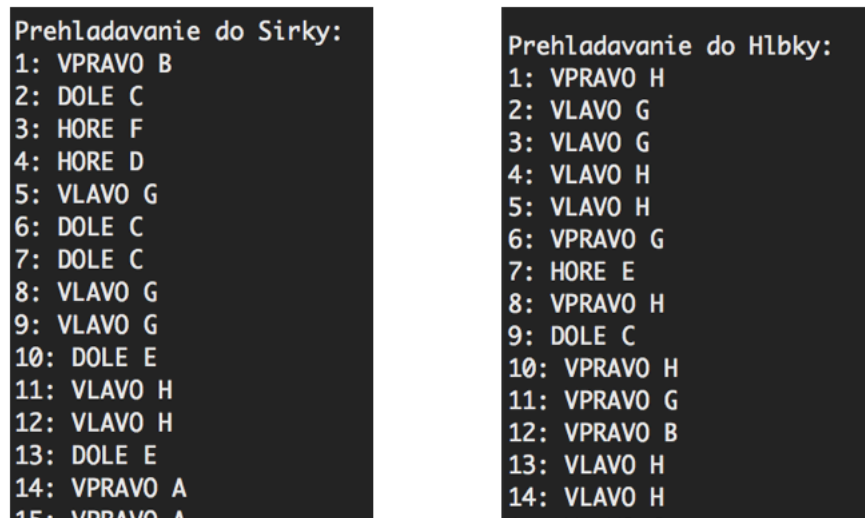
Prvé zadané autíčko reprezentuje „červené“ auto. Teda to, ktoré sa snažíme dostať napravo.

Každému auto sa prideli písmeno v abecede. Prvé (to ktoré sa má dostať do cieľa) je reprezentované písmenom A (1221), ďalšie B (0021), ďalšie C ...

Výstupom programu je výpis na terminál a zápis do súboru „blazniva\_krizovatka.txt“

Výpis na terminál vyzerá nasledovne:

Ako prvé sa zobrazia kroky k výsledku nájdené prehľadávaním do šírky a následne do hĺbky:



```

Prehľadavanie do Šírky:
1: VPRAVO B
2: DOLE C
3: HORE F
4: HORE D
5: VLAVO G
6: DOLE C
7: DOLE C
8: VLAVO G
9: VLAVO G
10: DOLE E
11: VLAVO H
12: VLAVO H
13: DOLE E
14: VPRAVO A
15: VPRAVO A

Prehľadavanie do Hĺbky:
1: VPRAVO H
2: VLAVO G
3: VLAVO G
4: VLAVO H
5: VLAVO H
6: VPRAVO G
7: HORE E
8: VPRAVO H
9: DOLE C
10: VPRAVO H
11: VPRAVO G
12: VPRAVO B
13: VLAVO H
14: VLAVO H
  
```

Obrázok 3 Ukážka výstupu na terminál

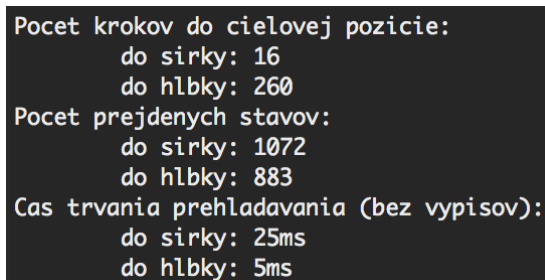
Kroky sú zobrazené v tvare : „X: Y Z“

X -> je číslo poradia posunu

Y-> smer akým sa dané auto pohlo

Z-> písmeno auta ktoré sa posúva

Ako posledný sa zobrazí výpis, ktorý hovorí o počte krokov do cieľa pri použití jednotlivých pozícií, počte prejdenných stavov ktoré boli kontrolované a čas trvania prehľadávania.



```

Pocet krokov do cielovej pozicie:
  do sirky: 16
  do hlbky: 260
Pocet prejdennych stavov:
  do sirky: 1072
  do hlbky: 883
Cas trvania prehladavania (bez vypisov):
  do sirky: 25ms
  do hlbky: 5ms
  
```

Obrázok 4 posledná časť výpisu na terminál

Tieto informácie užívateľ môže využiť na porovnanie jednotlivých algoritmov.

Výpis do súboru vypíše ako prvý počiatočný stav v grafickej mape:

```
Vasa pociatocna pozicia vyzerata takto:

BB---C
F--E-C
FAAE-C
F--E--
D---GG
D-HHH-
-----
```

Obrázok 5 výpis do súboru

Autá sú reprezentované písmenami, voľné miesta pomlčkou.

Následne ukáže postup do cieľa cez prehľadávanie do šírky a po ňom prehľadávanie do hĺbky:

<pre>Prehľadavanie do Šírky: 1 VPRAVO B  -BB--- F--E-C FAAE-C F--E-- D---GG D-HHH- -----  2 DOLE C  -BB--- F--E-C FAAE-C F--E-C D---GG D-HHH- -----</pre>	<pre>Prehľadavanie do Hĺbky: 1 VPRAVO H  BB---C F--E-C FAAE-C F--E-- D---GG D-HHH- -----  2 VLAVO G  BB---C F--E-C FAAE-C F--E-- D---GG- D-HHH- -----</pre>
---	---

Obrázok 6 Ukážka výstupu do súboru

Vo výpise do súboru je grafická mapa každého kroku a aj cesta v tvare ako je vo výpise na termináli.

## Testovania

Svoj program som testoval postupne, najskôr som testoval jednoduchá mapy ktoré som si overil ručne ako scenáre A,B. Neskôr som skúšal aj stavy ktoré nemajú riešenie scenáre C,D alebo stavy ktoré sú zložitejšie (z Obr.1) scenár E. Medzi posledným som skúšal mapy zo stránky: [http://cs.ulb.ac.be/~fservais/rushhour/index.php?window\\_size=20&offset=0](http://cs.ulb.ac.be/~fservais/rushhour/index.php?window_size=20&offset=0) kde som si overoval podľa počtu krokov či je môj algoritmus správny. Napríklad hneď prvý vstup scenár F

(výstupy uvádzam len terminálové nakoľko sú rovnaké ako v súbore len neobsahujú mapové zobrazenie):

Scenár A)

Vstup:

1  
2221

Výstup :

Prehľadavanie do Sirky:

1: VPRAVO A  
2: VPRAVO A

Prehľadavanie do Hlbky:

1: VPRAVO A  
2: VPRAVO A

Pocet krokov do cielovej pozicie:

do sirky: 2  
do hlbky: 2

Pocet prejdennych stavov:

do sirky: 5  
do hlbky: 4

Cas trvania prehľadavania (bez vypisov):

do sirky: 1ms  
do hlbky: 0ms

Scenár B)

Vstup

3  
2221  
0030  
5030

Výstup:

Prehľadavanie do Sirky:

1: VPRAVO A  
2: DOLE C  
3: DOLE C  
4: DOLE C  
5: VPRAVO A

Prehľadavanie do Hlbky:

1: DOLE C  
2: DOLE C  
3: DOLE C  
4: DOLE B  
5: DOLE B  
6: HORE C  
7: HORE C  
8: HORE C

9: DOLE B

10: VPRAVO A

11: DOLE C

12: DOLE C

13: DOLE C

14: VPRAVO A

Pocet krokov do cielovej pozicie:

do sirky: 5  
do hlbky: 14

Pocet prejdennych stavov:

do sirky: 41  
do hlbky: 40

Cas trvania prehľadavania (bez vypisov):

do sirky: 0ms  
do hlbky: 0ms

Scenár C)

Vstup:

9  
1221  
0021  
5030  
0420  
3020  
3230  
0130  
4421  
2531

Výstup:

Prehľadavanie do Sirky:

Neexistuje riesenie

Prehľadavanie do Hlbky:

Neexistuje riesenie

Pocet krokov do cielovej pozicie:

do sirky: 0  
do hlbky: 0

Pocet prejdennych stavov:

do sirky: 52  
do hlbky: 52

Cas trvania prehľadavania (bez vypisov):

do sirky: 2ms  
do hlbky: 1ms



Scenár D)

Vstup:

2  
2221  
4221

Výstup:

Prehľadavanie do Sirky:  
Neexistuje riešenie

Prehľadavanie do Hlbky:  
Neexistuje riešenie

Pocet krokov do cieľovej pozície:  
do sirky: 0  
do hlbky: 0

Pocet prejdenných stavov:  
do sirky: 6  
do hlbky: 6

Cas trvania prehľadavania (bez  
vypisov):  
do sirky: 0ms  
do hlbky: 0ms

Scenár E)

Vstup:

8  
1221  
0021  
5030  
0420  
3130  
0130  
4421  
2531

Výstup:

Prehľadavanie do Sirky:  
1: VPRAVO B  
2: DOLE C  
3: HORE F  
4: HORE D  
5: VLAVO G  
6: DOLE C  
7: DOLE C  
8: VLAVO G  
9: VLAVO G  
10: DOLE E  
11: VLAVO H  
12: VLAVO H  
13: DOLE E

14: VPRAVO A

15: VPRAVO A

16: VPRAVO A

(prehľadavanie do šírky malo veľa  
stavov, pre vypísanie spustíte daný  
vstup v programe.)

Pocet krokov do cieľovej pozície:  
do sirky: 16  
do hlbky: 260

Pocet prejdenných stavov:  
do sirky: 1072  
do hlbky: 883

Cas trvania prehľadavania (bez  
vypisov):  
do sirky: 29ms  
do hlbky: 4ms

Scenár F)

Vstup:

13  
2221  
0120  
1420  
2320  
3020  
4030  
5030  
0031  
1121  
0321  
2521  
4421  
4521

Výstup:

(pre lepšiu ukážku zobrazím iba koniec  
výpisu, nakoľko je tam veľmi veľa  
krokov)

Pocet krokov do cieľovej pozície:  
do sirky: 93  
do hlbky: 3120

Pocet prejdenných stavov:  
do sirky: 19349  
do hlbky: 13301

Cas trvania prehľadavania (bez  
vypisov):  
do sirky: 203ms  
do hlbky: 74ms

## Zhodnotenie

Tento projekt je podľa môjho názoru úspešný a jeho riešenie je dostatočne rýchle. Môj algoritmus bez akýchkoľvek problémov vyriešil aj zložitejšie zadania. Možné vylepšenie by bolo prerobiť ho na vstupnú mapu inú ako 6x6. Toto rozšírenie je implementované, avšak užívateľ nemá možnosť zmeny, nakoľko to nebolo v zadaní. Prehľadávanie do hĺbky má výhodu oproti prehľadávaniu do šírky v tom, že skôr nájde riešenie čo sa týka času a prejdejších stavov. Avšak prehľadávanie do šírky nájde to najkratšie riešenie, čo môžeme vidieť napríklad pri testovaní scenára F. Kde síce do hĺbky našlo riešenie skoro 3 krát rýchlejšie a prešlo skoro o tretinu stavov menej, no v konečnom dôsledku nájdené riešenie bolo skoro 34 krát dlhšie ako pri prehľadávaní do šírky.