Računalništvo 1: 1. izpit

28. januar 2021

Čas reševanja je 90 minut. Veliko uspeha!

1. naloga (15 točk)

Dan je sklad `Sklad_A`, na katerem so padajoče zložena števila (največji podatek je na dnu sklada). Primer:

```
Sklad_A = dno : 658 : 128 : 67 : 55 : 32 : 11 : 8 : 3 : -5 : -22
```

Na voljo imamo še prazen sklad `Sklad_B` in prazno vrsto `Vrsta`.

Opišite postopek, kako števila iz `Sklad_A` premaknemo v `Vrsta` tako, da bodo na začetku padajoče urejena soda števila, nato pa naraščajoče urejena liha števila. Na koncu postopka naj bosta oba sklada prazna. V postopku ni dovoljeno uporabljati dodatnih podatkovnih struktur. Če postopek uporabimo na zgornjem zgledu, dobimo:

```
Vrsta = začetek : 658 : 128 : 32 : 8 : -22 : -5 : 3 : 11 : 55 : 67 : konec
```

2. naloga (25 točk)

Potrebujemo podatkovno strukturo za pomnjenje elementov, ki pripadajo različnim kategorijam. Odločili smo se, da bomo uporabili *vrsto poimenovanih skladov*, kjer vrsta vsebuje elemente oblike `(ime, sklad)`. Zahtevamo, da je vrsta **urejena** glede na imena skladov (urejenost elementov v skladih ni pomembna). Primer:

	4		
128	65		
4	23		0
21	78		12
začetek : ("KR", dno) :	("LJ", dno) :	("MB", dno) :	("NM", dno) : konec

Podrobno opišite implementacijo spodnjih metod (pazite na urejenost). Za vsako od metod določite in argumentirajte tudi časovno zahtevnost v O notaciji, kjer naj bo n število vseh elementov v vseh skladih, m število skladov v vrsti in k velikost največjega (glede na število elementov) sklada v vrsti.

(a) `dodaj_sklad(ime)` v vrsto vstavi prazen sklad z izbranim imenom. Predpostavite lahko, da `ime` še ni uporabljeno.

(b) `vstavi_na_dno(ime, x)` poskusi na *dno* sklada z imenom `ime` vstaviti element `x`, vrne pa logično vrednost, ki opisuje uspeh (vrne `False`, če sklada z imenom `ime` ni v vrst oziroma `True`, če je postopek uspel). Metoda naj ima *prostorsko* zahtevnost $O(k)$.

(c) `vsote_skladov()` vrne *novo* vrsto, ki vsebuje pare `(ime, vsota)`, kjer je `vsota` seštevek vseh elementov sklada z imenom `ime`.

(d) `preimenuj(staro_ime, novo_ime)` preimenuje izbrani sklad (če obstaja).

Vedno lahko predpostavite, da se imena skladov ne ponavljajo.

3. naloga (20 točk)

Na voljo imamo *iskalno drevo s ponovitvami*, kjer za vsako vozlišče (x je vrednost podatka v vozlišču) velja:

- vsi podatki v levem poddrevesu so manjši ali enaki x;
- vsi podatki v desnem poddrevesu so večji ali enaki x.

Opišite algoritem, ki sprejme določeno vrednost in vrne število ponovitev te vrednosti v drevesu.

4. naloga (20 točk)

Psička Nara je na drugem koncu travnika opazila prečudovito palico. Do palice želi priti čim hitreje, vendar je travnik poln zanimivih vonjav, ki jo ovirajo pri hitrejšem napredovanju.

Travnik predstavimo z matriko, kjer na vsakem polju označimo, koliko časa Nara potrebuje, da preduha tisti košček travnika.

$i \setminus j$	0	1	2	3	4	5
0	1	2	1	2	2	1
1	3	6	0	2	7	0
2	0	1	3	4	8	3
3	2	5	7	0	1	4

Nara se nahaja v levem zgornjem kotu, palica pa v desnem spodnjem kotu. Pri tem se lahko psička premika desno, dol, ali pa diagonalno desno-in-dol. Opišite učinkovit algoritem, ki vrne poljubno pot, po kateri si Nara lahko prilasti palico v najkrajšem možnem času. Pot predstavimo kot seznam parov (i, j), ki predstavljajo indekse za polja v matriki.

Za zgornji travnik je primer optimalne poti:

```
[(0, 0), (0, 1), (1, 2), (2, 2), (3, 3), (3, 4), (3, 5)]
```

Algoritem naj bo ustrezno utemeljen, po možnosti z ustrezno Bellmanovo enačbo.

5. naloga (20 točk)

Rešujemo problem trgovskega potnika na omrežju:

$$\begin{bmatrix} \infty & 3 & 2 & 1 & 7 \\ 4 & \infty & 2 & 3 & 3 \\ 1 & 5 & \infty & 2 & 6 \\ 2 & 1 & 3 & \infty & 2 \\ 7 & 2 & 5 & 4 & \infty \end{bmatrix}$$

(a) Koliko stane krožna pot `1-4-3-5-2-1`?

(b) Podajte spodnjo mejo cene optimalne krožne poti za zgornje omrežje po metodi reduciranja matrike. Uporabite lahko poljuben vrstni red redukcij (najprej stolpci nato vrstice ali obratno).

(c) Kakšna je ocena vrednosti krožnih poti (po metodi reduciranja matrike), če zahtevamo, da krožna pot vsebuje segment `1-3-2`?