# Algoritmi in podatkovne strukture – 2
# Prvi kolokvij (2017/18)

Kolokvij morate pisati posamič. Pri reševanju je literatura dovoljena. Pri odgovarjanju bodi natančni in: (i) odgovarjajte *na zastavljena* vprašanja; in (ii) odgovorite na *vsa* zastavljena vprašanja – če boste odgovarjali na vsa vprašanja, lahko dobite dodatne točke.

Čas pisanja kolokvija je 60 minut.

Veliko uspeha!

| NALOGA | TOČK | OD TOČK | NALOGA | TOČK | OD TOČK |
|---|---|---|---|---|---|
| 1 | | | 3 | | |
| 2 | | | 4 | | |

IME IN PRIIMEK: ______________________

ŠTUDENTSKA ŠTEVILKA: ______________________

DATUM: ______________________

PODPIS: ______________________

**1. naloga:** *Paroma disjunktne množice.* Peter je tokrat dobil univerzalno množico $U = \{1, ..., n\}$ in učinkovite implementacije funkcij `Makeset()`, `Union()` in `Find()`, kot smo jih opisali na predavanjih. V pomoč, amortizirana časovna zahtevnost posamezne funkcije je bila $O(\log^* n)$.[1]

VPRAŠANJA:

A) Naj bo $n = 6$ in imejmo naslednji niz klicev funkcij:

`Makeset(5)`,`Makeset(1)`,`Makeset(2)`,`Makeset(4)`,`Makeset(6)`, `Makeset(3)`, `Find(5)`, `Union(5,1)`, `Union(2,3)`, `Find(5)`, `Union(1,2)`, `Union(3,4)`, `Union(6,6)`,`Find(5)` in `Find(1)`.

(i) Narišite podatkovno strukturo po vsakem klicu funkcije. (ii) Za vsakega od klicev opišite, katere primerjave ste naredili in koliko jih je bilo.

B) Amortizirana časovna zahtevnost niza $m$ klicev funkcij je definirana kot skupna poraba časa za vse klice in nato deljena s številom klicev funkcij $m$. Seveda, slednje pomeni, da je lahko eden od klicev zelo „drag“, drugi pa bistveno cenejši. (i) Opišite, kako izgleda podatkovna struktura, pri kateri pride do tega „dragega“ klica? Utemeljite odgovor. (ii) Kako izgleda zaporedje klicev funkcij, ki privedejo do podatkovne strukture v prejšnjem delu vprašanja? Utemeljite odgovor.

C) Spet Petrov šef. Tokrat želi poleg implementacije običajnih funkcij še funkcijo `Members(x)`, ki vrne seznam članov množice, kateri pripada element `x`. Časovna zahtevnost vaše rešitve naj bo čim boljša in vse točke dobite, če bo sorazmerna $O(k + \log^* n)$, kjer je $k$ število vrnjenih elementov.

**2. naloga:** *Slovar.* Recimo, da imamo množico števil $\{1, 2, ..., n\}$.

VPRAŠANJA:

A) (i) Najprej naj bo $n = 7$ in vstavite po vrsti elemente od $1$ do $7$ v AVL drevo. Izrišite drevo po vsakem vstavljanju. (ii) Opišite, kako izgleda na koncu drevo, če vstavite elemente od $1$ do poljubno velikega $n$.

B) (i) Narišite AVL drevo višine 5 (če imamo samo en element, je takšno drevo višine 1), ki ima najmanj elementov. (ii) Opišite, kakšno naj bo zaporedje operacij, ki ustvari takšno drevo.

[1] V knjigi je omenjena amortizirana časovna zahtevnost posamezne operacije $O(\alpha(m, n))$, kjer je $m$ število operacij. Ta funkcija v resnici raste še počasneje kot $\log^* n$.

C) Na predavanjih smo govorili o brisanju elementov iz AVL drevesa in profesor je narobe trdil, da je pri brisanju potrebno samo eno popravljanje (vrtenje). (i) Narišite drevo in zapišite, katero brisanje v njem zaheva več popravljanj. (ii) Koliko popravljanj je lahko potrebnih v drevesu višine $h$? Utemljite odgovor, se pravi, narišite drevo, v katerem so potrebna ta popravljanja.

**3. naloga:** Obstaja cela vrsta izvedb vrste s prednostjo. Značilnost vseh je, da lahko dostopamo do najmanjšega elementa v času $O(1)$. V tej nalogi bomo imeli opravka z vrsto s prednostjo, v kateri je $n > 2018$ elementov, kjer je $n$ naravno število.

VPRAŠANJA:

A) Recimo, da imamo izvedbo z dvojiško (binarno) kopico. V njej iščemo $k = 4$ najmanjši element. Kje vse se lahko nahaja? Utemeljite odgovor.

B) Kaj pa, če imamo binomsko kopico. Kje vse se lahko nahaj sedaj $k = 4$ najmanjši element? Ponovno utemeljite odgovor.

C) Vračamo se k dvojiški kopici. V prvem vprašanju očitno ni bilo potrebno preiskati celotne kopice, ko smo iskali $k = 4$ najmanjši element. Koliko mora biti *vsaj* $k$, kot funkcija $n$, da bomo morali preiskati celotno kopico? Utemeljite odgovor.

**4. naloga:** Imamo naslednjo definicijo števila $C_n$

$$C_n = n + \frac{1}{n} \sum_{1 \le i \le n} (C_{i-1} + C_{n-1}) \ . \tag{1}$$

V tej nalogi učinkovitejša rešitev dobi več točk.

VPRAŠANJA:

A) Izračunajte $C_5$ iz enačbe (1) in pri tem pokažite postopek računanja.

B) i.) Zapišite funkcijo (algoritem), ki izračuna $C_n$ iz enačbe (1) z uporabo dinamičnega programiranja, pri čemer uporabite rekurzijo. ii.) Kakšna je časovna in prostorska zahtevnost vaše rešitve. Utemeljite odgovor?

C) i.) Sedaj pa zapišite funkcijo (algoritem), ki prav tako izračuna $C_n$ iz enačbe (1) z uporabo dinamičnega programiranja, vendar gradi rešitev od spodaj navzgor. ii.) Kakšna je časovna in prostorska zahtevnost vaše rešitve. Utemeljite odgovor.