Algoritmi in podatkovne strukture – 2
Prvi kolokvij (2016/17)

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: *Slovar – osnove.* Recimo, da imamo naslednja števila, ki so urejena po velikosti: 1, 4, 19, 19, 19, 24, 39, 40, 42, 46, 50, 53, 56, 58, 59, 61, 66, 69, 74, 77, 79 in 90.

VPRAŠANJA:

A) Narišite najplitvejše urejeno dvojiško drevo, ki vsebuje vse zgornje elemente.

B) (i) Napišite algoritem, ki iz n *različnih* urejenih števil v polju `A` zgradi najplitvejše urejeno dvojiško drevo. Utemeljite, da je vaše drevo res najplitvejše. (ii) Kakšna je časovna zahtevnost vašega algoritma? Utemeljite odgovor.

C) Predlagajte podatkovno strukturo (ali strukture), ki bo porabila najmanj časa za naslednje zaporedje operacij: (i) najprej vstavimo n elementov (`Insert`); in (ii) nato naredimo n^2 poizvedb (`Find`). Koliko časa porabi vaša struktura (ali strukture) za vse operacije? Utemeljite odgovor.

NAMIG: Boljša kot bo vaša rešitev, več točk boste dobili.

2. naloga: *Slovar – nadgradnja.* Peter Zmeda dela domačo nalogo in je dobil naslednjih 18 števil v naključnem vrstnem redu:

$$69, 74, 58, 4, 61, 90, 19, 46, 1, 19, 40, 59, 39, 56, 19, 50, 42, 53. \quad (1)$$

Ker bo števila po vrsti vstavljal v preskočni seznam, potrebuje še naključne vrednosti 0 in 1:

1 0 1 0 1 0 0 1 0 1 0 1 1 0 1 0 1 0 0 0 1 0 1 0 0 0 0 1 0 1 0 0 0 1 0 1 1
1 0 0 0 0 0 1 1

VPRAŠANJA:

A) Kako izgleda Petrov preskočni seznam po vstavljanju števil? Upoštevajte, da je višina elementa število zaporednih enic plus 1: konkretno, prvi vstavljeni element, ki je 69, bo imel višino 2, saj eni enici sledi ničla (podčrtani prva 1 in 0 zgoraj). Pri vstavljanju naslednjega elementa, ki je 74, uporabite naslenjo enico in ničlo ter tako naprej.

B) Definirajmo nad slovarjem funkcijo `Left(x)`, ki vrne največji element, ki je v slovarju in je manjši od elementa `x` – vrne levega soseda elementa `x`. Peter mora implementirati novo funkcijo. Pomagajte mu.

(i) Opišite, kako naj deluje nova funkcija[1]. (ii) Kakšna je časovna zahtevnost vašega algoritma? Utemeljite odgovor. (iii) Bi lahko nadgradili (spremenili) preskočni seznam, da bi pohitrili iskanje levega elementa? Utemeljite odgovor.

[1] Psevdokoda ali resničen algoritem prinaša več točk.

C) Petrov učitelj je neizmeren vir idej, ki jih mora nato Peter implementirati. Tokrat se je spomnil, da bi implementacijo slovarja s preskočnim seznamom nadomestil z razpršilno funkcijo, ker je nekje prebral, da naj bi bila slednja učinkovitejša. (i) Kako naj sedaj Peter implementira funkcijo `Left(x)`? (ii) Kakšna je časovna zahtevnost implementacije? Utemeljite odgovor. (iii) Kaj pa sedaj, se da kaj narediti, da bi bila implementacija hitrejša? Kaj pa v primeru, če že imamo referenco na element `x` ali, če ga ni v slovarju, na prvega večjega? Utemeljite odgovor.

3. naloga: Recimo, da imamo univerzalno množico števil $\{1, 2, ..., 16\}$ in vsako od njih predstavlja ločeno samostojno množico.

VPRAŠANJA:

A) Nad množico množic simulirajte operacije: `Union(1, 7)`, `Union(2, 9)`, `Union(3, 9)`, `Union(15, 10)`, `Union(16, 1)`, `Union(12, 12)` in `Union(13, 7)`. Zapišite množice, ki ste jih dobili na koncu.

B) In spet Petrov učitelj. Tokrat želi, da mu Peter implementira podatkovno strukturo, ki bo podpirala poleg operacij `Union()` in `Find()`, ki smo ju spoznali na predavanjih pri disjunktnih množicah, še operacijo `List(x)`, ki vrne vse elemente, ki pripadajo množici, katere pripadnik je tudi `x`. (i) Opišite novo podatkovno strukturo. (ii) Kakšna je časovna zahtevnost vsake od treh funkcij? Utemeljite odgovor.

C) Pri disjunktnih množicah velja, da nek element pripada natančno eni množici. Recimo, da tokrat vsak element pripada največ dvema množicama. (i) Opišite ustrezno podatkovno strukturo, ki bo podpirala učinkovito operaciji `Union()` in `Find()`. (ii) Opišite obe operaciji ter utemeljite njuno pravilnost in časovno in prostorsko zahtevnost. (iii) Ali opazite kakšen vsebinski in konceptualni problem definicije tega problema ter kako bi se ga lotili?

NAMIG: Kaj pravzaprav vrne `Find()`?

4. naloga: *Vrste s prednostjo, izbira in rang.*

VPRAŠANJA:

A) Vrnimo se k 18 naključnim številom v eq. (1). (i) Prvih 10 števil po eno vstavite v leno binomsko kopico H_1. Narišite kopico H_1 po tem vstavljanju in zapišite koliko operacij ste naredili pri vsakem vstavljanju in koliko vse skupaj. (ii) V drugo leno binomsko kopico H_2 vstavite preostalih 8 števil. Narišite kopico H_2 po vstavljanju in zapišite število operacij, ki ste jih naredili tokrat pri posameznem vstavljanju in vse skupaj. (iii) Izvedite operacijo `DelMin(`H_2`)`, narišite kopico po zaključku operacije in zapišite število izvedenih operacij. (iv) Na koncu še zlijte kopici H_1 in H_2 ter ponovno narišite končno kopico ter zapišite število operacij, ki ste jih opravili tokrat.

   NAMIG: V kopici morate vedno vedeti, kateri element je najmanjši.

B) Pri iskanju mediane smo razdelili množico vhodnih podatkov na peterke. (i) Ali bi lahko razdelili na sedmerke – bi algoritem še deloval? Utemeljite odgovor. (ii) Kaj pa, če razdelimo na enajsterke – bi algoritem še deloval? Utemeljite odgovor tokrat.

C) Rang malce drugače. Peter Zmeda ima n_1 podatkov že urejenih po velikosti v polju. Po tem dobi n_2 novih podatkov, ki jih spravlja v razpršeno tabelo. Na koncu pride še zahteva, naj izračuna `Rank(x)`. (i) Opišite postopek in podatkovno strukturo, kako naj to naredi učinkovito. (ii) Utemeljite pravilnost vaše podatkovne rešitve. (iii) Ocenite časovno in prostorsko zahtevnost vaše rešitve.

   NAMIG: Rezultat bo vseboval kot parametra n_1 in n_2.