Vaje APS2 / 2022 / 04
Teden: 7. – 11. Mar. 2022

Naloga 1

Popolnoma dopolnjen seznam je kazalčni seznam, v katerem vsak element vsebuje kazalec na naslednika (torej na element +1), vsak drugi element vsebuje kazalec na element +2, vsak četrti element vsebuje kazalec na element +4 in tako naprej.

a) Koliko kazalcev vsebuje popolnoma dopolnjen seznam z desetimi elementi?
b) Najmanj koliko in največ koliko primerjanj bi morali narediti, če bi iskali element v popolnoma dopolnjenem seznamu s 100 elementi?
c) Katere kazalce bi morali spremeniti za vzdrževanje strukture, če bi v popolnoma dopolnjenem seznamu z desetimi elementi pobrisali 7. element? Odgovor zapiši kot seznam kazalcev takole: če je, na primer, potrebno spremeniti kazalec, ki kaže iz tretjega elementa na njegovega naslednika, zapiši 3(+1).

Naloga 2

Imamo preskočni seznam, ki ga bomo implementirali z uporabo stražarja (sentinel) za iskanje in vstavljanje. Vloga stražarja je, da poenostavi kodo tako, da ni potrebno neprestano preverjati, ali je referenca na naslednji element NULL.

a) Narišite poljubni preskočni seznam s primernim stražarjem in z elementi 7, 3, 6, 10 in 9. Utemeljite svojo izbiro stražarja!
b) Recimo, da je preskočni seznam s stražarjem na začetku prazen; izvedemo naslednje zaporedje klicev funkcij:

   vstavi(5), vstavi(2), vstavi(1), vstavi(12), vstavi(11), vstavi(18), vstavi(10), vstavi(21), **briši(11)**, **briši(1)**, **vstavi(23)**, **vstavi(13)**.

   Generator naključnih števil vrača naslednje zaporedje (pravilo določanja nivojev je takšno: vsak element je že po definiciji na nivoju 1, nato pa nivoje elementu morebiti višamo dokler nam generator vrača enico):

   1101010000111010001100011101100010101011111000101010000100111 10…

   Zapišite stanje preskočnega seznama pred in po poudarjenih klicih briši() / vstavi() v zgornjem zaporedju.

Naloga 3

V tabeli tabel podrobneje obravnavamo predvsem operaciji *insert()* in *find()*, operacija ***delete()*** pa tipično ni podana. Ugotovili smo tudi že, da si lahko pri operaciji *find()* pomagamo z min/max informacijo, ki je naravno vsebovana v vsaki urejeni pod-tabeli.

Pri tej nalogi razmislite in podajte smiselno in učinkovito implementacijo operacije ***delete()***, ki ohranja »dobre« lastnosti tabele tabel glede na operaciji *insert()* in *find()* v smislu časovne zahtevnosti.