08.71.23/24 Tölvunarfræði 2/2a

Upptökupróf **20. ágúst, 2003**
Kennari: Hjálmtýr Hafsteinsson **kl. 9^{00} – 12^{00}**

Fyrstu 5 dæmin eru fyrir alla nemendur (bæði í Tölvunarfræði 2 og 2a) Dæmi 6 er aðeins fyrir nemendur í Tölvunarfræði 2, en dæmi 7 er aðeins fyrir nemendur í Tölvunarfræði 2a (verkfræði-nema). Í báðum tilfellum gilda ***fimm bestu dæmin af sex****. Dæmin hafa jafnt vægi.*

Öll skrifleg hjálpargögn og reiknivél leyfð*.*

- Athugið að þegar beðið er um að "Lýsa" eða "Sýna" þá er nóg að gera það í orðum og með teikningum. Ef þið eigið að skrifa C++ kóda þá er beðið um það sérstaklega.
- Rökstyðjið öll svör og munið að það er óþarfi að skrifa upp skilgreiningar sem eru í bókinni.

1. Við lausn þessa dæmis eigið þið að nota ykkur skilgreininguna á stærðargráðu (e. Big-Oh notation) á bls. 44 í kennslubókinni.

a) Gildir að $N^{1.1} + N \log N$ er $O(N \log N)$?

b) Gildir að $N^k / \log N$ er $O(N^k)$?

c) Gildir að 2^{aN} er $O(2^N)$, þar sem a er fasti?

2. Í útfærslunni á biðröð (e. queue) með eintengdum lista sem sýnd er í kennslubókinni á bls. 169 eru notaðir bendarnir `head` og `tail`, sem benda fremst og aftast í tengda listanum. Útskýrið hvernig hægt er að útfæra biðröð, sem er jafn hraðvirk, með einum bendi, köllum hann `p`, ef eintengdi listinn er hringtengdur.

Sýnið útfærslu í C++ á `QUEUE` klasanum, sem notar aðeins bendinn `p`.

3. Tvíundartré er einkvæmt ákvarðað af *forröðun* (e. preorder) og *innröðun* (e. inorder) þess. Lýsið nákvæmlega **endurkvæmu** (e. recursive) reikniriti sem fær inn forröðun og innröðun tvíundartrés, og býr til tréð sjálft. Gerið ráð fyrir að reikniritið fái inn tvo vektora, `pre` og `in`, ásamt fjölda staka í þeim, N. Það búi síðan til hnútana í trénu og skili út bendi á rót trésins sem vektorarnir lýsa. Lýsing ykkar á að vera það nákvæm að auðvelt sé að færa hana yfir í C++ kóda. Þið megið auðvitað líka skila þessu sem C++ falli, en þá með góðum útskýringum.

4. Gefin eru N gildi (k_1, k_2, ..., k_N) sem á að setja inní tvíleitartré (e. binary search tree). Eftir að tvíleitartréð hefur verið búið til þá verður leitað í því að þessum N gildum. Tíðni mismunandi leitanna er þekkt. Þannig verður leitað p_1 sinnum að staki k_1, p_2 sinnum að staki k_2, o.s.frv. Athugið einnig að lögun lögun tvíleitartrésins ræðst af röð innsetninganna, þegar tréð er búið til.

a) Það er augljóst að við viljum hafa þau stök sem oft er leitað að, ofarlega í trénu. Er alltaf best að hafa stakið með hæstu tíðnina sem rótina? Rökstyðjið eða sýnið einfalt mótdæmi.

b) Lýsið gróflega aðferð til að nota þessa vitneskju um leitartíðnina til að ákveða lögun tvíleitartrésins, með það fyrir augum að lágmarka heildarleitartíma.

5. Hægt er að hugsa sér hrúgu (e. heap) setta upp sem *þríundartré* í stað tvíundartrés eins og gert er í venjulegu útgáfunni.

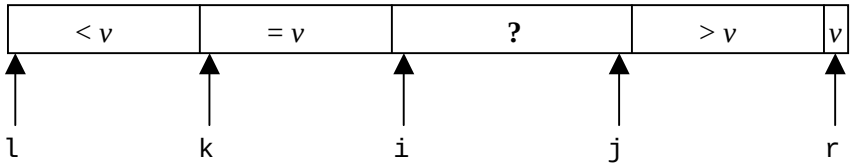
a) Sýnið hvernig hægt er að ferðast upp og niður í trénu með einföldum reikniaðgerðum á vísa vektorsins, svipað og gert er í tvíundartrés-útgáfunni.

b) Útfærið í C++ föllin `fixUp` og `fixDown` á sambærilegan hátt og gert er á bls. 384 og 385 í kennslubókinni, ef litið er á hrúguna sem þríundartré.

Aðeins fyrir nemendur í Tölvunarfræði 2:

6. Í kafla 7.6 í kennslubókinni er sýnd aðferð til að skipta stökum vektors uppí þrjú svæði: "minni en *v*", "jöfn *v*" og "stærri en *v*". Aðferðin í bókinni setur stökin sem eru jöfn *v* fremst og aftast í vektorinn meðan á aðferðinni stendur (eins og lýst efst á bls. 337), en víxlar þeim síðan inn að miðju í lokin.

Þið eigið að útfæra aðra aðferð sem notar aðeins aðra leið til að skipta vektornum í þrjú svæði. Hægt er að lýsa aðferðinni með eftirfarandi mynd:



Útfærið aðferðina sem fall í C++, svipað og gert er í forriti 7.2 á bls. 319 í kennslubók. Útskýrið einnig nánar hvernig aðferðin virkar og berið hana saman við aðferðina í kafla 7.6 (sem er útfærð á bls. 338).

Aðeins fyrir nemendur í Tölvunarfræði 2a (verkfræðinema):

7. Í þeirri línulegri hökkun (e. linear probing) sem lýst er í Kafla 14.3 í kennslubókinni er stakið *v* sem verið er að setja inn, sett í næsta auða hólf á eftir hólfinu sem það á að fara í, ef það hólf er upptekið. Það er hægt að breyta þessari hegðun á ýmsa vegu. Lýsið kostum og göllum eftirfarandi aðferða miðað við aðferðina í bókinni og hvenær þær eru betri og hvenær verri.

a) Í stað þess að færa *v* niður hakkatöfluna ef það er árekstur, þá er *v* sett í rétt hólf, en stakið sem var fyrir í því hólfi sett í næsta auða hólf töflunnar.

b) Í stað þess að leita einungis að auðu hólfi niður töfluna, þá er leitað að næsta auða hólfi fyrir ofan **eða** neðan hólfið, ef hólfið sem *v* á að koma í er upptekið.

c) Í stað þess að athuga sífellt næsta hólf fyrir neðan í leit að auðu hólfi þá er notuð fyrirfram ákveðin talnaröð, sem er alltaf sú sama, t.d. hólfið + 25, síðan hólfið + 4, þá hólfið + 81, o.s.frv. Öll þessi gildi eru auðvitað módúlus stærð töflunnar.