TÖL203G Tölvunarfræði 2

Heimadæmi 6

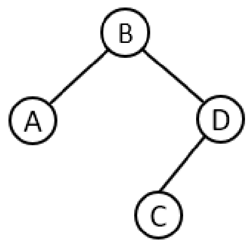
Í síðustu viku var farið í kafla 3.1 um táknatöflur og 3.2 um tvíleitartré. Í næstu viku verður tekin fyrir ein gerð af tvíleitartrjám í jafnvægi: rauð-svart tré úr kafla 3.3.

Heimadæmin eru **til þess að þjálfa ykkur** í efninu – nýtið þau vel! Einkunn fyrir þau mun **ekki lækka lokaeinkunn**, þannig að þið fáið mun meira út úr því að glíma við dæmin sjálf en að fá aðstoð frá Hr. Google.

Æfingadæmi fyrir dæmatíma 27. og 28 feb.

1. Á hve marga vegu er hægt að setja inn lyklana A, B, C, D, þannig að þeir myndi tvíleitartréð hér til hliðar? Sýnið alla möguleikana.

2. Er Hibbert eyðing (`delete`) úr tvíleitartré samhverf? Hvert er lokatréð eftir að lyklunum A og B hefur verið eytt úr trénu hér til hliðar? Annars vegar þegar A er eytt fyrst og síðan B, og hins vegar þegar B er eytt fyrst og síðan A.



Heimadæmi (skila í Gradescope)

1. [*Óraðaður listi*] Þið eigið að breyta táknatöfluútfærslunni SequentialSearchST.java, þannig að listinn sé sjálfskipandi (*self-organizing*). Sjálfskipandi gagnagrindur laga sig að notkunarmynstri notandans, þannig að lyklar sem oft er leitað að finnast hraðar en þeir sem sjaldan er leitað að. Þið eigið að útfæra tiltekna útgáfu sem kallast Færa-fremst (*move-to-front*). Hún felst í því að þegar kallað er á `get(k)`, þá er hnúturinn með lyklinum k færður fremst í tengda listann (ef hann finnst). Það þýðir að ef leitað er aftur að k fljótlega þá finnst hann hratt.

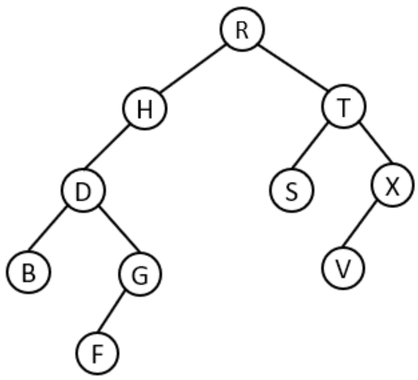
   Þið eigið að skila breytta fallinu `get` og skjáskoti af keyrslu á `main`-fallinu fyrir inntakið "A B R A C A D A B R A", sem þið sláið inn eða pípið úr skrá. Útkoman ætti að vera "D 6, C 4, R 9, B 8, A 10", þ.e. sætisnúmerin á síðasta tilvikinu af hverjum staf.

2. [*Raðað fylki*] Dæmi 3.1.28 á bls. 392 í kennslubók. Það á að breyta fallinu `put` í klasanum BinarySearchST.java þannig að ef nýr lykill er stærri en allir lyklarnir í töflunni þá er hann settur inn á föstum tíma (þ.e. $\Theta(1)$ í stað $\Theta(\log N)$). Skilið breytta fallinu `put` og skjámynd af keyrslu á inntakinu "A B C D E F G H".

3. [*Tvíleitartré*] Gefið er tvíleitartréð hér til hliðar. Notið Hibbard eyðingu (eins og sýnd er í bókinni og á glærunum) til þess að eyða lyklum út úr þessu tré:

   a. Eyða lyklinum "H" úr trénu. Sýnið hvaða hnúta aðferðin skoðar og teiknið upp lokatréð.

   b. Eyða lyklinum "D" úr upphaflega trénu. Sýnið hvaða hnúta aðferðin skoðar og teiknið upp lokatréð.



4. [*Tvíleitartré*] Notið áfram upphaflega tvíleitartréð í dæmi 3. Teljið upp hnútana sem skoðaðir eru þegar eftirfarandi tvíleitartrésaðgerðir eru framkvæmdar og gefið skilagildi fallsins:

   a. `ceiling("D")`

   b. `select(7)`

   c. `rank("T")`

   d. `floor("E")`

5. [*Tvíleitartré*] Í þessu dæmi eigið þið að skoða hversu há tvíleitartré verða á slembnu inntaki. Ljúkið við klasann MeasureBST.java sem býr til n-staka tvíleitartré með `Double` lykli og `Integer` gildi. Lykilgildið er fengið með `StdRandom.uniformDouble()` og `Integer` gildið getur verið hvað sem er. Í hverri tilraun (*trial*) er fundin hæð tvíleitartrésins og lokaniðurstöður forritsins eru meðalhæð tvíleitartrjánna. Forritið á líka að reikna út bestu mögulegu hæð tvíleitartrés með *n* stök (sem er $\lfloor \log_2 n \rfloor$) og prenta út hversu miklu hærri slembitrén eru miðað við besta mögulegt. Skilið klasanum `MeasureBST` og skjáskoti af keyrslu með n = 100 000 og 10 tilraunum. Hér fyrir neðan er dæmi um keyrslu með *n*=10 og 10 tilraunir:

```
E:\Forrit\hd6>java MeasureBST 10 10
For n = 10, optimal height is 3
Average height in 10 trials is 4.60,  1.53 times optimal
```

Skilið PDF-skjali með lausnum ykkar á þessum dæmum fyrir **kl. 23:59 fimmtudaginn 2. mars** í Gradescope. Munið eftir að gefa upp á hvaða blaðsíðum svör við einstökum dæmum eru.