# TÖL203G Tölvunarfræði 2

## Heimadæmi 5

Vegna miðmisserisprófsins þá eru þessi heimadæmi úr efni þriggja fyrirlestra. Það eru kaflar 2.2 um Mergesort, 2.3 um Quicksort og kafli 2.4 um forgangsbiðraðir (þ.e. hrúgur). Kafli 2.5 er samantekt um notkun á röðun, sem flest hefur komið fram í fyrirlestrunum. Í næstu viku byrjum við á kafla 3 um táknatöflur (3.1) og tvíleitartré (3.2).

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 20. og 21. feb.**

1. Við byrjum með tóma forgangsbiðröð og gerum eftirfarandi aðgerðir á hana (tala þýðir innsetning á tölunni og * þýðir `delMax`-aðgerðin): `8 3 2 * 10 7 * 5 12 4 * * * 9 * * 6 * * *`. Sýnið röð talanna sem `delMax`-aðgerðin skilar.

2. Stærsta stakið í hrúgu (*heap*) hlýtur að vera í sæti 1 í fylkinu. Næststærsta stakið hlýtur að vera í sæti 2 eða sæti 3. Segjum að við höfum 15-staka hrúgu *H*.
   a. Í hvaða sætum getur <u>þriðja</u> stærsta stakið verið í *H*?
   b. Í hvaða sætum getur <u>þriðja</u> stærsta stakið **ekki** verið í *H*?
   c. Hvað með <u>fjórða</u> stærsta stakið?

**Heimadæmi (skila í Gradescope)**

1. [*Mergesort*] Skoðið Java kóðann fyrir Mergesort (Algorithm 2.4 í bókinni, glæra 8 í fyrirlestri 9). Segjum að við köllum aðeins á `merge`-fallið (neðsta línan í `sort`-fallinu) ef `a[mid+1]` er minna en `a[mid]`.
   a. Hvers vegna er í lagi að gera þetta?
   b. Á hvernig inntaki myndum við græða mest á að bæta þessu inn?

2. [*Quicksort*] Leysið dæmi 2.3.18 á bls. 305 í kennslubókinni. Kallið ykkar útgáfu `QuickX` og berið hana saman við upphaflegu útgáfuna í bókinni (Quick.java) með forritinu SortCompare.java. Þið getið hent út úr `SortCompare` notkun á öðrum röðunar-aðferðum. Skilið breyttu útgáfunni af `partition`-fallinu og niðurstöðu úr samanburði á `Quick` og `QuickX` í `SortCompare`. Til að fá raunhæfan samanburð notið `n` = 1000000 og `trials` = 10. Þá ættuð þið líka að breyta útprentunarskipuninni í `SortCompare`, þannig að þið fáið fleiri aukastafi í hlutfallinu milli tímanna. Það ætti ekki að vera mjög mikill munur á þessum tveimur útgáfum. Hvers vegna er það?

3. [*Quicksort*] Fylkinu [2, 3, 1, 4, 5, 7, 6, 8] hefur verið skipt (*partitioned*) um vendistak (og það sett á réttan stað), en það er ekki gefið upp hvert vendistakið var. Hver af stökunum gætu hafa verið vendistakið? Teljið upp öll möguleg stök og rökstyðjið að þau séu möguleg og hin séu það ekki.

4. [*Hrúgur*] Gefið er fylkið [5, 4, 8, 1, 6, 3, 5, 7]. Sýnið hvernig það raðast með hrúguröðun (*heap sort*) með því að búa til mynd sem er sambærileg við myndina á bls. 324 í kennslubókinni (eða á glæru 35 í fyrirlestri 11). Sýnið líka hrúguna sem tvíundartré eftir að fylkinu hefur verið hrúguraðað (þrep 1).

5. [*Forgangsbiðraðir*] (Autograder) Dæmi 2.4.30 í kennslubók. Í þessu dæmi á að útfæra gagnagrindina `MedianFinder` með aðgerðirnar `insert(key)` sem setur inn lykilinn `key` á $\Theta(\log N)$ tíma, `deleteMedian()` sem skilar og eyðir út miðgildi stakanna sem eru í safninu á $\Theta(\log N)$ tíma. Loks er aðgerðin `findMedian()` sem skilar miðgildinu án þess að eyða því, á $\Theta(1)$ tíma. Þið fáið hér beinagrind að útfærslu: MedianFinder.java. Þið ættuð að útfæra þetta með því að nota tvær forgangsbiðraðir, eina min (þ.e. MinPQ.java) og eina max (þ.e. MaxPQ.java). Þið skilið þessu dæmi í *autograder* í Gradescope. Einkunnin fyrir það verður sameinuð við einkunnir fyrir önnur dæmi í þessum dæmaskammti.

   *Vísbending*: Það er ítarlegri vísbending um lausn þessa dæmi á heimasíðu kennslubókarinnar (þið þurfið að leita að því!).

---

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