## Litt om prioritetskøer og binære heaps — 10 poeng

Med *heap* mener vi arrayer som representerer binære min-heaps.

(a) En array med $n$ elementer kan gjøres om til en heap i $\mathcal{O}(n)$.

Sant.

(b) En min-heap blir en max-heap ved å reversere arrayet.

Usant.

(c) Alle elementer på dybde $d$ i en heap er mindre enn alle elementer på dybde $d+1$.

Usant.

(d) Innsetting i en heap med $n$ elementer er $\mathcal{O}(\log(n))$ i verste tilfelle.

Sant.

(e) Et AVL-tre kan brukes som en prioritetskø med samme kjøretidskompleksitet som en heap.

Sant.

(f) Nodene langs en sti fra rotnoden til en løvnode i en heap er ordnet fra minst til størst.

Sant.

(g) Dersom vi bytter plass på to søskennoder i en heap, er det fremdeles en heap.

Usant.

Oppgaven er litt tvetydig om hvorvidt man skal ta med seg subtrærne eller ikke. Tanken var å kun bytte om elementene i nodene. Selv hvis man tar med seg subtrærne har man ingen garanti for at det resulterende treet er *komplett*, og dermed er det heller ikke en heap. Siden begge tolkninger gir opphav til samme svar lar vi oppgaven stå.

(h) En prioritetskø med konstant tid på alle operasjoner ville gjort at Dijkstra hadde kjøretidskompleksitet $\mathcal{O}(|V|+|E|)$.

Sant.

(i) Man kan finne det *største* elementet i en *min*-heap i $\mathcal{O}(\log(n))$.

Usant.

(j) I en heap ligger *over halvparten* av elementene på de *to dypeste* nivåene.

Sant.