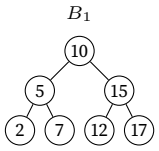
Intervall i et binært søketre 10 poeng

Du er gitt et binært søketre B og to heltall a og b der $a \leq b$. Du skal gi en algoritme som skriver ut verdiene innenfor intervallet $[a, b]$ i *sortert* rekkefølge (fra minst til størst). Det vil si at hvis x er verdien i en node, så skal denne noden skrives ut hvis og bare hvis $a \leq x \leq b$.

Input: Et binært søketre B og to heltall a og b der $a \leq b$
Output: Skriver ut verdiene innenfor intervallet $[a, b]$ i sortert rekkefølge

```
1 Procedure InRange(B, a, b)
  | // ···
```



For det binære søketreet B_1 (vist i eksempelet ovenfor) skal for eksempel `InRange(`B_1`, 2, 7)` skrive ut 2, 5 og 7, og `InRange(`B_1`, 10, 20)` skrive ut 10, 12, 15 og 17.

(a) Oppgi algoritmen din (med psuedokode). Du kan gjøre rimelige antagelser om representasjonen av B. En mer effektiv algoritme er mer poenggivende. Du trenger ikke tenke på formateringen på utskriften.

I de neste deloppgavene skal du anta at B har n noder, er balansert og ikke inneholder noen duplikater.

(b) Anta at a er den minste verdien i B og b er den største verdien i B. Oppgi kjøretidskompleksiteten på algoritmen din med hensyn til n.

(c) Anta at $a = b$. Oppgi kjøretidskompleksiteten på algoritmen din med hensyn til n.

Vi gir ingen føringer på hvordan det binære treet skal være implementert. Studenten må selv gjøre rimelige antagelser her (som at det en node har et element x en venstre- og en høyrepeker). Her antar vi at B er representert ved rotnoden, og tar denne noden som input. Hver node har en venstre, høyre og en x. Vi representerer det tomme treet ved **`null`**.

(a)

Input: Et balansert binært søketre B og to heltall a og b der $a \leq b$
Output: Skriver ut verdiene innenfor intervallet $[a, b]$ i sortert rekkefølge

```
1 Procedure InRange(v, a, b)
2   if v = null then
3     return null
4   if a < v.x then
5     InRange(v.left, a, b)
6   if a ≤ v.x ≤ b then
7     print(v.x)
8   if v.x < b then
9     InRange(v.right, a, b)
```

Denne algoritmen har kjøretidskompleksitet $O(\log n + \min(n, (b - a)))$, men dette trenger ikke studenten å oppgi.

(b) Algoritmen kjører i $O(n)$ dersom a og b er henholdsvis det største og minste tallet i B.

(c) Dersom $a = b$ er kjøretidskompleksiteten $O(\log n)$.

For full uttelling må studenten ha en god fremstilling av en algoritme som har $O(\log n)$ når $a = b$, og oppgi riktig kjøretidskompleksitet. En $O(n)$ algoritme kan få opp til 8 poeng. En $O(n \cdot \log n)$ algoritme kan få opp til 5 poeng. En $O(n^2)$ algoritme kan få opp til 2 poeng.