# Oblig3

Innhold

[Oblig3 1](#_Toc2145339312)

[Oppgave 1 2](#_Toc1624910092)

[A) 2](#_Toc1570733722)

[B) 2](#_Toc2144908648)

[Oppgave 2 2](#_Toc74933372)

[A) (“Ferdig”) 2](#_Toc549603838)

[B) (“Ferdig”) 2](#_Toc1685609885)

[Oppgave 3 (Ferdig) 3](#_Toc1438961880)

[A) (Ferdig) 3](#_Toc1203238483)

[B) (Ferdig) 3](#_Toc1223168387)

[B) (Ferdig) 4](#_Toc1288997348)

[Oppgave 4 4](#_Toc1076552430)

[A) (Ferdig) 4](#_Toc556991522)

[B) (Ferdig) 4](#_Toc1903347782)

[C) 4](#_Toc40682317)

[Oppgave 5 4](#_Toc2127416504)

[A) 4](#_Toc1127138115)

[B) (Ferdig) 4](#_Toc1622363853)

[C) (Feil) 5](#_Toc935440061)

[D) 7](#_Toc2031089313)

[E) (Ikke ferdig) 7](#_Toc1113495730)

[F) 8](#_Toc2108513734)

## Oppgave 1

### A)

### B)

## Oppgave 2

### (“Ferdig”)

Sortering ved innsetting (O(n^2))

|  |  |  |  |
| --- | --- | --- | --- |
| n | Antall målinger | Målt tid i sekunder (gjennomsnittlig) | Teoretisk tid (c\*T(n)) |
| 32000 | 50 | 0,0507 | c= |
| 64000 | 50 | 0,0510 |  |
| 128000 | 50 | 0,0511 |  |

Utvalgssortering (O(n^2))

|  |  |  |  |
| --- | --- | --- | --- |
| n | Antall målinger | Målt tid i sekunder (gjennomsnittlig) | Teoretisk tid (c\*T(n)) |
| 32000 | 50 | 0,0665 |  |
| 64000 | 50 | 0,0658 |  |
| 128000 | 50 | 0,0664 |  |

Kvikksortering (log2(n))

|  |  |  |  |
| --- | --- | --- | --- |
| n | Antall målinger | Målt tid i sekunder (gjennomsnittlig) | Teoretisk tid (c\*T(n)) |
| 32000 | 50 | 0,0020 |  |
| 64000 | 50 | 0,0017 |  |
| 128000 | 50 | 0,0023 |  |

Flettesortering (n\*log2(n))

|  |  |  |  |
| --- | --- | --- | --- |
| n | Antall målinger | Målt tid i sekunder (gjennomsnittlig) | Teoretisk tid (c\*T(n)) |
| 32000 | 50 | 0.0034 |  |
| 64000 | 50 | 0,0027 |  |
| 128000 | 50 | 0,0037 |  |

### (“Ferdig”)

|  |  |  |
| --- | --- | --- |
| n | Antall målinger | Målt tid i sekunder (gjennomsnittlig) |
| 32000 | 50 | 0,01 |
| 64000 | 50 | 0,016 |
| 128000 | 50 | 0,027 |

Kvikk sorter fungerer best om elementene er langt ifra hverandre, så får alle elementene er like, vil den uansett bruke tid på å sortere elementene selv om de er sorterte.

## Oppgave 3 (Ferdig)

### (Ferdig)

**Binært tre:**

Et tre er en graf hvor det ikke finnes noen sykler, det vil si at det ikke finnes en node hvor man kan gå i sirkel. Det betyr at alle noder skal ha en unik sti får å komme seg fra roten til den aktuelle noden. Et tre er bygget opp av foreldre (røtter i undertrær) og barn. Alle barn har nøyaktig en forelder, og en forelder kan ha et spesifikt maks antall barn, men et binært tre har ikke mer enn 2 barn.

**Høyden**:

Høyden til et tre er antall steg (noder) man må passere før man kommer til den siste noden i den lengste grenen.

**Et balansert binært tre**:

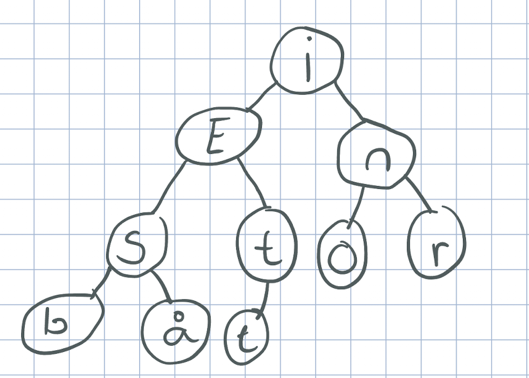
Et balansert tre er et tre hvor lengden på alle blad ikke har en differanse på mer enn 1.

**Et komplett binært tre:**

Et komplett tre er et tre hvor alle foreldre maks antall barn unntatt på det siste nivået hvor nodene er blad.

### B) (Ferdig)

#### i)



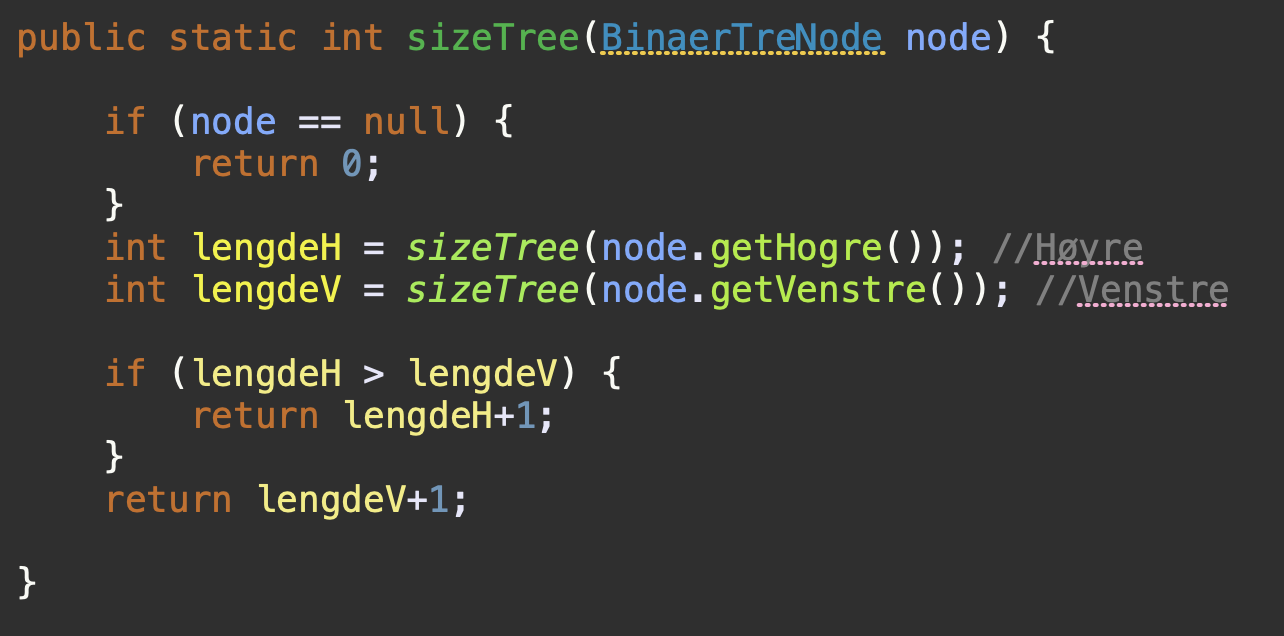
#### ii)

### (Ferdig)

|  |  |
| --- | --- |
| Tre 1 | Tre 2 |
| Pre-orden | |
| iesOtrntbå | iesOtrntbå |
|  |  |
| In-orden | |
| tOsreibåtn | etoSribåtn |
| Post-orden | |
| tOrsbåteni | tOrsebåtni |

## Oppgave 4

### (Ferdig)



### B) (Ferdig)

##### i-vi)

### C)

## Oppgave 5

### A)

**Haug:**

Ein haug er ein type binært tre. Denne kan brukes til å sortere data og lage ein prioritetskø.

I ein haug sorterer ein data utifrå det som har høgast prioritet. Dersom elementene har lik prioritet bruker køen “først inn, først ut” prinsippet.

### B) (Ferdig)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 9 | | | | | | | |
| 15 | | | | 12 | | | |
| 7 | | 4 | | 2 | | 1 | |
| 6 | 5 | 3 |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | | | | | | | |
| 12 | | | | 10 | | | |
| 11 | | 2 | | 6 | | 3 | |
| 4 | 8 | 1 |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | | | | | | | |
| 10 | | | | 14 | | | |
| 8 | | 7 | | 13 | | 6 | |
| 2 | 5 | 4 |  |  |  |  |  |

1. Største verdi til haugen skal være i noden, dette stemmer for tre b og c.

2. Foreldernoden skal alltid være større en barna, stemmer for b og c.

Tre b og c er hauger

### C) (Feil)

#### i)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | | | | | | | |
| 10 | | | | 14 | | | |
| 8 | | 7 | | 13 | | 6 | |
| 2 | 5 | 4 |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | | | | | | | |
| 10 | | | |  | | | |
| 8 | | 7 | | 13 | | 6 | |
| 2 | 5 | 4 |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | | | | | | | |
| 10 | | | | 13 | | | |
| 8 | | 7 | |  | | 6 | |
| 2 | 5 | 4 |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | | | | | | | |
| 10 | | | | **13** | | | |
| 8 | | 7 | |  | | 6 | |
| 2 | 5 | 4 |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | | | | | | | |
| 10 | | | | 13 | | | |
| 8 | | 7 | | **4** | | 6 | |
| 2 | 5 |  |  |  |  |  |  |

#### ii)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | | | | | | | |
| 10 | | | | 13 | | | |
| 8 | | 7 | | 4 | | 6 | |
| 2 | 5 | **13** |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | | | | | | | |
| 10 | | | | 13 | | | |
| 8 | | **13** | | 4 | | 6 | |
| 2 | 5 | **7** |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | | | | | | | |
| **13** | | | | 13 | | | |
| 8 | | **10** | | 4 | | 6 | |
| 2 | 5 | 7 |  |  |  |  |  |

### D)

Prinsipp **Makshaug:**

Hvordan kan ein makshaug brukes til å sortere elementer?

I ein makshaug sammmenligner du venstre og høgre element med kvarandre for å sortere i ein makshaug. Da slipper du å sammenligne alle elementene med hverandre.

### E) (Ikke ferdig)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | | | | | | | |
| 9 | | | | 8 | | | |
| 7 | | 6 | | 5 | | 4 | |
| 3 | 2 | 1 |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | | | | | | | |
| 9 | | | | 8 | | | |
| 7 | | 6 | | 5 | | 4 | |
| 3 | 2 | **10** |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | | | | | | | |
| **2** | | | | 8 | | | |
| 7 | | 6 | | 5 | | 4 | |
| 3 | **9** | 10 |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | | | | | | | |
| **2** | | | | 8 | | | |
| 7 | | 6 | | 5 | | 4 | |
| 3 | **9** | 10 |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | | | | | | | |
| **1** | | | | 8 | | | |
| 7 | | **9** | | 5 | | 4 | |
| 3 | 2 | 6 |  |  |  |  |  |

### F)