Finn k minste — 12 poeng

Gitt et array A med n sammenlignbare elementer, ønsker vi å finne de k minste elementene i A der $1 \leq k \leq n$.

Input: Et array A av n sammenlignbare elementer, og et heltall $1 \leq k \leq n$
Output: En liste med de k minste elementene fra A

```
1 Procedure FindMin(A, k)
  | // ···
```

Skriv en prosedyre som beregner `FindMin(`A`, `k`)` i:

> Merk at et korrekt svar på deloppgave (c) også er et korrekt svar på (a) og (b). Dermed kan man få full uttelling dersom man oppgir at svaret på (c) også er et svar på (a) og (b).
>
> Merk igjen at output for algoritmene under er en *liste*, men det er en liten detalj som ikke er vesentlig for *algoritmene*. Vi trekker ikke poeng for dette alene, men det kan være med på å trekke i samspill med andre unøyaktigheter.

(a) $\mathcal{O}(k \cdot n)$

> En naturlig algoritme i $\mathcal{O}(k \cdot n)$ er å finne det minste elementet, deretter finne det nest minste, og så videre, k ganger. En annen måte å se på det er å gjøre k iterasjoner av `SelectionSort`. Løsningen under er utledet fra `SelectionSort`.

```
1  Procedure FindMin(A, k)
2  |  list ← an empty list
3  |  for i ← 0 to k do
4  |  |  m ← i
5  |  |  for j ← i + 1 to n − 1 do
6  |  |  |  if A[j] < A[m] then
7  |  |  |  |  m ← j
8  |  |  append A[m] to list
9  |  |  A[i], A[m] ← A[m], A[i]
10 |  return list
```

- Inntil 3 poeng for en løsning i $\mathcal{O}(k \cdot n)$.
- Trekk 1 poeng dersom man ikke ser vekk fra elementer som allerede er lagt til i resultatet.
- Trekk 1 poeng dersom man ikke håndterer at to elementer kan være like.

(b) $\mathcal{O}(n \cdot \log(n))$

> Denne løsningen svarer til å sortere arrayet med en $\mathcal{O}(n \cdot \log(n))$ sorteringsalgoritme, og plukke ut de k første.

```
1 Procedure FindMin(A, k)
2 |  list ← an empty list
3 |  A ← MergeSort(A)
4 |  for i ← 0 to k do
5 |  |  append A[i] to list
6 |  return list
```

- Inntil 4 poeng for en korrekt løsning.
- Inntil 3 poeng dersom `Quicksort` benyttes (denne er i $\mathcal{O}(n^2)$, men er vanligvis rask).
- Inntil 1 poeng for løsninger som baserer seg på en kvadratisk sorteringsalgoritme.

(c) $\mathcal{O}(n \cdot \log(k))$ *eller* $\mathcal{O}(n + k \cdot \log(n))$

Problemet kan løses i $\mathcal{O}(n \cdot \log(k))$ ved å bruke en max-heap til å holde på de k minste elementene.

```
1  Procedure FindMin(A, k)
2      list ← an empty list
3      heap ← an empty max-heap
4      for i ← 0 to k − 1 do
5          add A[i] to heap
6      for i ← k to n − 1 do
7          heap.push(min(heap.pop(), A[i]))
8      for i ← 0 to k − 1 do
9          prepend heap.pop () to list
10     return list
```

Merk at man også kan anta at heapen har metode `peek`, og kun poppe dersom $A[i]$ er mindre enn det største elementet på heapen.

Problemet kan løses i $\mathcal{O}(n + k \cdot \log(n))$ ved å først bygge en min-heap i lineær tid (vi kjenner at vi kan bygge en max-heap på linær tid fra forelesning, og en min-heap er helt tilsvarende), deretter poppe de k minste elementene fra en heap med n elementer.

```
1  Procedure FindMin(A, k)
2      list ← an empty list
3      heap ← BuildMinHeap(A)
4      for i ← 0 to k − 1 do
5          append heap.pop () to list
6      return list
```

- Inntil 5 poeng for en korrekt løsning i $\mathcal{O}(n \cdot \log(k))$ *eller* $\mathcal{O}(n + k \cdot \log(n))$.