# Uge 1 opgaver

Jens Evald-Schelde, Jonathan Gabal Christiansen og Emil B. Henriksen

7. september 2018

**Abstract**

A review of this weeks exercises. Algorithms and pseudo code.

## 1 Del

Vi skal bestemme hvad algoritmen ud spytter med forskellige parameter.
Ved at gennemgå algoritmen med de givende parametre, kan vi bestemme om den returnerer True eller False.

```
1  exists(A,n,x):
2          lo = 0
3          hi = n-1
4          while hi >= lo:
5                  mid = floor((hi+lo)/2)
6                  if x > A[mid]:
7                          lo = mid+1
8                  else if x < A[mid]:
9                          hi = mid-1
10                 else:
11                         return true
12         return False
```

Linje 1 indeholder funktionens navn, og de parameter den tager. 'A' er en array, 'n' er arrayens længde og 'x' er den integer vi leder efter i arrayen.

På linje 2 sættes variablen 'lo' til 0, og på linje 3 bliver 'hi' sat til længden af arrayen minus en.

På linje 4 startes vores while loop, der kører så længe at forholdet er sandt, hvilket er at 'hi' er højere eller lig 'lo'. Hvis det ikke skulle være tilfældet returneres False.

På linje 5 bliver variablen 'mid' sat til 'hi' + 'lo' divideret med 2, og så rundet ned, f.eks. floor(2.5) == 2, floor(2.9) == 2.

På linje 6 bliver 'x' tjekket om den er større end værdien som er på arrayens indeksplads svarende til værdien i variablen 'mid'. Hvis det er sandt, bliver 'lo' sat til 'mid + 1' og loopet gentages med de nye værdier. Hvis 'x' er mindre end 'A[mid]' køres else if blokken, 'hi' bliver sat til 'mid - 1', og loopet gentages.

Hvis de to værdier skulle være lige store, vil det sidste forhold blive kørt, som returnere True på linje 11. Hvis loopet aldrig finder det ønskede element, eller hvis der ikke søges igennem hele længden, vil der returneres False.

**a** exists(A, 8, 17)
returns true

**b** exists(A, 8, 9)
returns false

**c** exists(A, 4, 12)
returns false

**d** mids værdier med parameterne (A,8,2)

| loop | antal |
|---|---|
| 0 | 3 |
| 1 | 1 |
| 2 | 0 |
| 3 | return false |

# 2 Del

Algoritmen er ment til at gennemkøre en array, for at se om 'x' er tilstede i arrayen. Parameteren n giver et startspunkt for algoritmen, og er samtidigt længden på arrayen. Grundet algoritmens opbygning vil den også tjekke 'bag sig' hvis den værdi der tjekkes er større end det søgte. En af grundende til at algoritmen fungerer, er, at den allerede er sorteret fra mindste til højeste værdi. En usorteret array ville give et udfald der ikke er forudsigeligt eller efter hensigten, da den er baseret på en sorteret array af positive integers $\mathbb{Z}^+$ altså A[0]<=A[...]<=A[n-1].

# 3 Del

(a) Vi kan ikke lige se et tilfælde hvor True ville blive returneret, uden at integeren findes i arrayen. En falsk positiv burde altså ikke være mulig.

(b) Lad os antage at Arrayen A var sorteret omvendt, altså
A[0]>=A[...]>=A[n-1] A=[43, 17, 16, 12, 10, 6, 5, 1]

Når funktionen exists() kaldes med parametrene som i forrige del, vil den forsøge at finde x på den forkerte side, og derfor aldrig finde frem til x der rykker længere og længere væk.

# 4 Del

Hvis vi antager at en array er 64 elementer lang, der alle er sat til at være 0, så vil den i de værste tilfælde højest køre 7 loops log2(64)+1. Grunden hertil, er at værdien i 'hi' ikke bliver påvirket, og 'lo' forøges ved hver iteration af while loopet. Variablen mid vil altså antage følgende værdier [31, 47, 55, 59, 61, 62, 63] modsat [31, 15, 7, 3, 1, 0] hvis der ledes i starten af arrayen.