TDDC91,TDDE22,725G97 Lektion 1

Ordo-notation och tidskomplexitet

Magnus Nielsen
magnus.nielsen@liu.se

6 september 2018

Regler

Några regler från Fö1 som kan vara användbara.

- (Ordo) $f \in O(g)$ omm det existerar $c > 0, n_0 > 0$ sådana att $f(n) \leq c \cdot g(n)$ för alla $n \geq n_0$
  Intuition: Bortsett från konstanta faktorer växer f inte snabbare än g

- (Omega) $f \in \Omega(g)$ omm det existerar $c > 0, n_0 > 0$ sådana att $f(n) \geq c \cdot g(n)$ för alla $n \geq n_0$
  Intuition: Bortsett från konstanta faktorer växer f minst lika fort som g

- (Theta) $f(n) \in \Theta(g(n))$ omm $f(n) \in O(g(n))$ och $g(n) \in O(f(n))$
  Intuition: Bortsett från konstanta faktorer växer f och g lika snabbt.

  NOTERA: Ω är motsatsen till O, dvs $f \in \Omega(g)$ omm $g \in O(f)$.

Övningsuppgifter

- Uppgift 1.
  Vilken eller vilka av följande förslag är ekvivalenta med $O(n^3)$?
  $O(n^3 + n \log n)$
  $O(14n^3)$
  $O(n^3 + 3)$
  $O(n^3 - n^2)$
  $O((n^2)(n + 4))$
  $O(n(n^2 - 5))$

- Uppgift 2.
  Vilken tidskomplexitet har följande funktion?

```
int n_fakultet(int const n) {
  int ans{1};
    for (int i = n; i > 0; i--) {
        ans = ans * i;
    }
    return ans;
}
```

- Uppgift 3.
  Vilken tidskomplexitet har följande funktion?

```
int calc(int n) {
  int ans{};
    for (int i = 0; i < n*n; i++) {
        for (int j = 0; j < n-3; j++) {
            ans += i+j;
        }
    }
    return ans;
}
```

- Uppgift 4.
  Visa att funktionen f(n) = $2n^3$ + 3n + 18 $\in \Theta(n^3)$.

  Håll i åtanke reglerna från Fö1.