Datastrukturer & Algoritmer, Datalogi C
Henning Christiansen15/9-2001

Opgaver til forelæsningen 17/9-2002

Der er ikke øvelsesgang 17/9 (OB) og 20/9 (Bas), men brug opgaverne som hjemmearbejde og evt. på øvelserne senere.

Afleveringsopgave for DatC: Opgave 3 nedenfor; afleveringsfrist 1/10 (form og sted aftales med Lars)

Opgave 1
Bogens "bevis" for sætning 5.1 er mildest talt obskurt. Vis ved traditionelle induktionsbeviser følgende:

1) $1 + 2 + ... + n = n(n+1)/2$
2) $1 + (1+2) + (1+2+3) + ... + (1+2+...+n) = n(n+1)(n+2) / 6$

Opgave 2
Omformulér bogens beskrivelser af Ω, Θ og o (lille 0) vha. af grænseværdier analogt til den måde vi i forelæsningen definerede O:

En funktion T(n) er O(F(n)) såfremt $T(n)/F(N) \rightarrow$ konstant når $n \rightarrow \infty$

Bogens opgave 5.7 (side 177) spg. a og b.

Opgave 3
Opgave handler om repræsentation af mængder af tal ved et array udstyret med en tæller; vi skitserer det som en klasse således:

```
public class talmængde{
  public indsæt(int x) {... indsætter x i mængden };
  public boolean med_i(int x) {... returnerer true hvis x med i mængden, false ellers};
  private int [] indhold = new int [1000000];
  private int antal = 0
}
```

Problemer hvis arrayet løber fuldt skal ignoreres.

Implementér klassen i to udgaver

1) Tallene indsættes i den rækkefølge de ankommer (dvs. svarende til rækkefølgen af kald af indsæt); med_i metoden implementeres som sekventielt gennemløb
2) Indholdet af arrayet holdes til enhver tid sorteret. Dette opnås ved at indsæt fungerer således:
    a. først opsøges index hvor det nye tal skal placeres (f.eks. ved binær søgning)

b. alle elementer til højre herfor skubbes en position mod højre, så der bliver plads til det nye tal.

Metoden for med_i kan implementeres ved binær søgning.

Angiv O for metoderne med_i og indsæt for begge metoder; hvad siger det om, i hvilke situationer implementation 1 eller 2 er mest velegnede.