Datavetenskap I

Föreläsning 9

Problemlösning

Babylon 5

Ceremonies of Light and Dark

Föreläsning 9

- Innehåll
  - Föreläsning 9
    - Pseudokod (halvkod)
    - Problemlösning
    - Algoritm
    - Selektion
    - Iteration
  - Läsanvisningar
- Påminnelse för programstudenter
  - Drivhuset tisdag eller onsdag beroende på grupp. Info finns på P1-sidorna på It's L.

Pseudokod

- Programspråk är formella språk där vi måste hålla oss till vissa regler.
- Pseudokod är ett informellt sätt att skriva riktlinjerna för hur ett stycke kod ska se ut.
- Används för att testa om en lösning kommer att fungera eller för att skissa upp kod innan vi vet detaljer om koden.

Pseudokod

- Använder naturligt språk men strukturerat
- Indrag viktiga – visar delar som hör ihop i ett block
- Bryt ner problem till lämplig detaljnivå
- Aktivitetsdiagram kan användas som en form av pseudokod men kallas oftast inte pseudokod.

Problemlösning

- Problemlösning
  - Samla information om problemet
  - Analysera och omformulera problemet
  - Undersöka olika möjliga lösningar
  - Utvärdera lösningar/ bestämma sig för en lösning
  - Implementera lösningen
- Vilket är det minsta problem vi kan lösa enskilt?

Analysen är viktigt – vad är egentligen problemet, omformulering kan behövas för att få ett greppbart problem.
Bryt eventuellt ner problemet i flera mindre problem

Vad är analys och vad är design i punkterna ovan?

Algoritm

- En lösning för ett problem
- Beskriver hur vi vill styra flödet i koden
- Kan återanvändas när samma eller liknande problem dyker upp igen.

Selektion

- Enkel selektion:
  OM (villkor utvärderas till sant)
      utför det här blocket
- Selektion med alternativ:
  OM (villkor utvärderas till sant)
      utför det här blocket
  ANNARS
      utför det här alternativa blocket

Hur kan vi rita detta med aktivitetsdiagram?

Läs in två tal från användaren
OM (tal1 > tal2)
  skriv ut ”tal 1 är störst”
ANNARS
  skriv ut ”tal 2 är störst”

Vad händer om talen är lika stora?

Selektion

- Vanliga satser för selektion i Java:
  - if-sats
  - if-else-sats
  - switch-sats
- Fler typer av selektioner finns och ni kommer att stöta på dem efter hand.

Block i Java
```
{
   Kod i blocket
}
```

Switch-sats:
JÄMFÖR (variabel med värde)
  om värdet är X: gör det här och avbryt
  om värdet är Y: gör det här och avbryt
  om ingen match: gör det här

Nästla

- Vi kan skriva en ny selektion inuti blocket som hör till en selektion:
  OM (villkor utvärderas till sant)
    gör någonting
    OM (annat villkor som utvärderas till sant)
      gör det här
    utför en sats till

```
OM (tal1==tal2)
   skriv ut ”talen är lika stora”
ANNARS
   OM (tal1 > tal2)
      skriv ut ”tal 1 är störst”
   ANNARS
      skriv ut ”tal 2 är störst”
```

Iteration

- Iteration:
  SÅ LÄNGE (villkor är sant)
  utför det här blocket
- Kallas ofta loop
- Loopar kan bli oändliga om villkoret inte kan bli sant - programmet låser sig.
- Ofta finns en räknare som styr loopen

Hur kan vi rita detta med aktivitetsdiagram?

SÅ LÄNGE (räknare < antal gånger i loopen)
  gör något
  öka räknare med 1

Iteration

- Vanliga satser för iteration i Java
  - For-loop
  - While-loop
  - Do-while-loop
- Fler typer av iterationer finns och du kommer att stöta på dem senare.

While-loopen utförs när vi inte vet hur många gånger vi behöver upprepa något
Do-while-loopen används när vi inte vet hur många gånger vi behöver upprepa något men vet att det SKA ske MINST EN gång.
For-loopen används när vi vet hur många gånger vi vill göra något, kombineras med en räknare

Nästla

- Loopar kan nästlas men en loop inuti en annan loop.
- SÅ LÄNGE (villkor är sant)
  SÅ LÄNGE (annat villkor är sant)
  gör det här

```
Sätt räknare1 till 0
SÅ LÄNGE (räknare1 < 3)
   Sätt räknare2 till 0
   SÅ LÄNGE (räknare2 < 2)
      skriv ut (räknare 1:räknare 2)
      öka räknare 2 med 1
   öka räknare1 med 1
```

Vad skrivs ut?

0:0
0:1
1:0
1:1
2:0
2:1

Nästla

- Selektioner och iterationer kan nästlas med varandra på alla möjliga sätt.

```
Sätt räknare till 0
Sätt antalRättAnv1 till 0
Sätt antalRättAnv2 till 0
Sätt antalRundor till 10
SÅ LÄNGE (räknare < antalRundor)
   läs in tal från användare1 i tal1
   läs in tal från användare2 i tal2
   OM (tal1==tal2)
      skriv ut ”Rundan blev oavgjord”
   ANNARS
      OM (tal1 > tal2)
         skriv ut ”Användare 1 vann rundan”
          öka antalRättAnv1 med 1
      ANNARS
         skriv ut ”Användare 2 vann rundan”
         öka antalRättAnv2 med 1
   öka räknare med 1

OM (antalRättAnv1==antalRättAnv2)
   skriv ut ”Matchen blev oavgjord”
ANNARS
   OM (antalRättAnv1 > antalRättAnv2)
      skriv ut ”Användare 1 vann matchen”
   ANNARS
      skriv ut ”Användare 2 vann matchen”
```

Läsanvisningar

- Java Foundations
  - Kapitel 1.3: Läs detta igen och tänk över vad problemlösning innebär inom mjukvaruutveckling.
  - Kapitel 4: Kan läsas igenom efter denna föreläsning. Efter övriga föreläsningar denna vecka kan kapitlet läsas noggrant då ni behöver kunna använda selektioner och iterationer i er kod.
- Extra materiel
  - Det finns en text med ett extra exempel på kursplatsen.