Rekursion

I denna laboration kommer vi att titta lite närmare på hur rekursion fungerar.

Mål

Du ska efter denna laboration kunna:

- förstå hur rekursion går till
- använda rekursion
- förstå när man skall undvika rekursion
- veta vad begreppet ändrekursion betyder
- använda iteration i kombination med rekursion

Tonvikt läggs på:

- struktureringen av problemet
- läsbarhet av programkod

Uppgift 1 (frivillig)

Du skall skriva den rekursiva implementationen av funktionen som har deklaration enligt följande:

```
function Factorial(N : in Natural) return Positive;
```

Funktionen skall beräkna N! utifrån det N som kommer in som parameter. För att visa att funktionen fungerar krävs förstås ett huvudprogram som anropar funktionen.

Uppgift 2 (frivillig)

Du skall skriva de rekursiva implementationen av funktionen som har deklaration enligt följande:

```
function Power(X : in Float;
               N : in Integer) return Float;
```

Funktionen skall beräkna X^N.

Uppgift 3

Du skall skriva en funktion som beräknar det N:te talet i Fibonacci-serien. Indata till funktionen skall vara N och funktionen skall givetvis vara rekursiv. Funktionen skall heta Fib.

Definition av Fibonacci-serien:

```
Fib(1) = 1
Fib(2) = 1
Fib(N) = Fib(N - 1) + Fib(N - 2)
```

Serien börjar alltså som följer:

```
1 1 2 3 5 8 13 21 34 ...
```

Testa med ett stort tal på N och se vad som händer. Varför blir det så?

Uppgift 4

Skriv ett program som läser in ett antal rader med tre tecken långa strängar. När inmatningen ”---” kommer skall alla tidigare inmatningar skrivas ut i omvänd ordning. Det kan vara godtyckligt många rader att mata in (vilket gör att du inte kan lagra dem i ett fördimensionerat fält).

Exempel på programkörning:

```
Mata in ett antal rader (3 tecken per rad, avsluta med ”---”):
aaa
bbb
ccc
---
Du matade in följande rader:
ccc
bbb
aaa
```

Uppgift 5

Som alla vet kan velociraptorer öppna dörrar. Detta tar dock lite tid så man kan tjäna på att stänga dörrar efter sig när man flyr för sitt liv. Första dörren tar 5 minuter att öppna. För varje dörr därefter tar det halva tiden av den föregående (enligt www.velociraptors.info).

Du har 9 minuter och 13 sekunder kvar till det att räddningen kommer. Hur många dörrar skall du passera (och stänga efter dig) för att klara dig undan velociratorns vrede?

Skriv ett progam som räknar ut hur många dörrar du skall passera. Programmet skall skriva ut dels antalet dörrar och dels hur lång tid du fick på dig för detta antal dörrar.

Krav: Programmet skall lösa uppgiften rekursivt.

Körexempel (som kanske inte har rätt antal dörrar och kanske ej heller rätt tid):

```
Det krävs 4 dörrar och
det gav dig 9 minuter och 23 sekunders flykttid.
```

OBS! Vi anger alltid tiden i minuter och hela sekunder (avrundat) i utskriften.

Tips: Man kan behöva tre parametrar som skickas i rekursionen (antal dörrar, tid som gått åt och tid att öppna nästa dörr).

P.S. Vem räddade dig? En T-rex? D.S.