

1) Datorer och program

Dator - bearbetar data (information)

Program - anger hur bearbetningen ska göras

- jämför program med recept

↓

Dator som

förstår 0

och 1.

↓

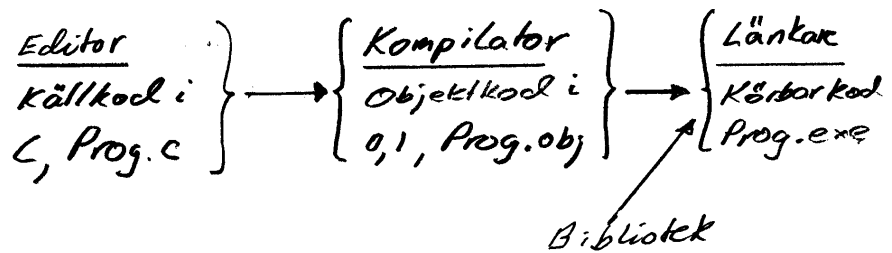
Människa

som förstår

svenska

- att skriva ett program till en dator kan jämföras med att skriva ett recept med enbart orden ja och nej. Innebär kodning!
svårt? Det finns hjälpmedel!

Programutveckling - för att konstruera ett program använder man editor, kompilator, länkar enligt



①

- (Ex) Skriv ett program i C som läser in ett reellt tal från tangentbordet och skriver ut talets kvadratroten på skärmen.

Körexempel

Ge ett reellt tal : 1.44 Matas in från tangentbordet med ett avslutande ENTER.
Roten : 1.20

Program

/* Rot.c */ En kommentar som kompilatorn hoppar över.

```
#include <stdio.h> // bibliotek, in- och utmatning  
#include <conio.h> // skärmhantering  
#include <math.h> // Matematik bibliotek
```

```
void main() // Huvudprogram! Körning börjar  
{ // alltid här!
```

```
    /* Variabler */  
    float tal, rot;
```

```
    /* Läs tal från tangentbordet */
```

```
    printf("Ge reellt tal : "); // skriver ledtext  
    scanf("%f", &tal); // läser tal och stoppar  
    // det i minnesplats som
```

```
    (2) // har samma adress som tal
```

```

/* Beräkna roten */
rot = sqrt(tal);

/* Skriv roten */
printf("Roten: %.2f", rot); // %.2f 2 decimaler

/* Vänta på tangenttryck */
getch();
}

```

Indentering - indrag med 3 blanka efter klammer
 { och tillbaka efter }

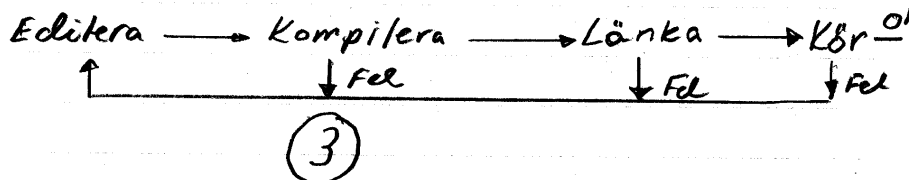
Små- och stora bokstäver - kompilatorn ställer på
 små och stora bokstäver.

```

void main
{
  → float tal, Tal, ... // Indrag 3 blanka
    |                   // Tal och tal är två olika
  }                     // variabler.

```

Testning - testkör alltid programmet med
 ett antal olika värden



Ex) Ändra i programmet Rot.c så att ett felmeddelande skrivs ut om det inmatade talet är negativt.

```
/*Rotneg.c*/
```

```
#include <stdio.h>  
#include <conio.h>  
#include <math.h>
```

```
void main()
```

```
{
```

```
float tal, rot;
```

```
printf("Ge ett positivt reellt tal: ");  
scanf("%f", &tal); // Formatspecifikation %f
```

```
if (tal >= 0.0) // om tal positivt
```

```
{
```

```
rot = sqrt(tal);
```

```
printf("Roten: %.2f", rot);
```

```
}
```

```
else
```

```
// annars
```

```
{
```

```
printf("Kan ej ta roten av negativt tal!");
```

```
}
```

```
getch()
```

```
}
```

④

(Ex) Ändra i programmet så att det upprepar inläsningen och rotberäkningen tills 0.0 inmatas.

Kör exempel

Ge ett positivt reellt tal (avslut 0.0): 1.44

Roten: 1.20

Ge ett positivt reellt tal (avslut 0.0): 49

Roten: 7.00

Ge ett positivt reellt tal (avslut 0.0): 0.0

/* Rotupp.c */

```
#include <stdio.h>
```

```
#include <math.h>
```

```
void main()
```

```
{
```

```
    float tal, rot;
```

```
    printf("Ge ett positivt reellt tal (avslut 0.0): ");
```

```
    scanf("%f", &tal);
```

```
    while (tal != 0.0) // så länge tal inte är 0.0
```

```
    {
```

```
        if (tal > 0.0)
```

```
        {
```

(5)

```

    rot = sqrt(tal);
    printf("Roten: %0.2f\n", rot); // \n ny rad!
}
else
{
    printf("Kan ej beräkna roten av negativt tal!\n");
}
printf("Ge ett positivt reellt tal (avslut 0,0):");
scanf("%0f", &tal);
}
}

```

OBS! Uppreningen är en while-loop och villkoret för uppreningen kontrolleras alltid innan man gör in i loopen och man måste läsa in nytt tal sist i loopen

//Läs tal!

```

while (tal != 0.0)
{

```

//Läs tal!

```

}

```

OBS! Nytt värde till tal måste läsas in annars har tal samma gamla värde och vi får en oändlig loop.

(6)

I ovanstående exempel har vi använt de tre programstrukturer som alla program konstrueras med, nämligen programmeringens A, B, C

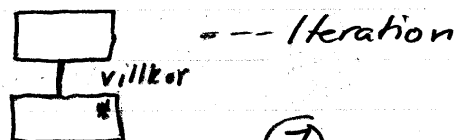
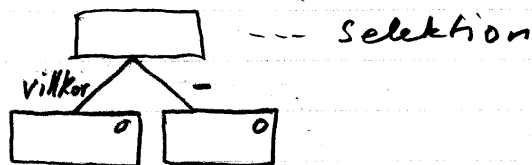
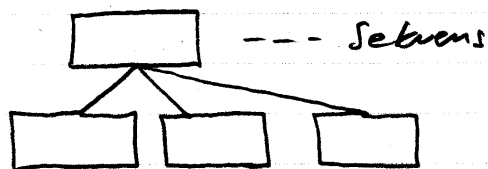
sekvens - först, sedan, sedan..... ordningsföljd

selektion - om, annars om..... val

iteration - noll, en, eller flera ----- uppreping

Genom att blanda dessa tre strukturer på ett korrekt sätt får man ett program som löser det aktuella problemet.

De tre strukturerna brukar grafiskt ritas i ett strukturiagram enligt



(7)

Ex) Skriv ett program som läser in två hela tal och skriv ut talen i storleksordning med det minsta först. Programmet ska upprepa och avsluta då första talet ges som 0.

```
/* Tal.c */
```

```
void main()
```

```
{
```

```
    int forsta, andra; // OBS! int (integer)
```

```
    printf("Ge första talet (avslut 0) : ");
```

```
    scanf("%d", &forsta); // %d för decimalt  
                        // heltal (10 bas)
```

```
    while (forsta != 0)
```

```
{
```

```
    printf("Ge andra talet : ");
```

```
    scanf("%d", &andra);
```

```
    if (forsta < andra)
```

```
{
```

```
        printf("%d \n %d", forsta, andra);
```

```
}
```

```
    else
```

```
{
```

```
        printf("%d \n %d", andra, forsta);
```

```
}
```

8


```

    printf("Ge första talet (avslut 0) : ");
    scanf("%d", &forsta);
  }
}

```

Test!

Ge första talet (avslut 0) : 2

Ge andra talet : 36

2

36

Kolla omvänt då andra < första

Ge första talet (avslut 0) : 56

Ge andra talet : 14

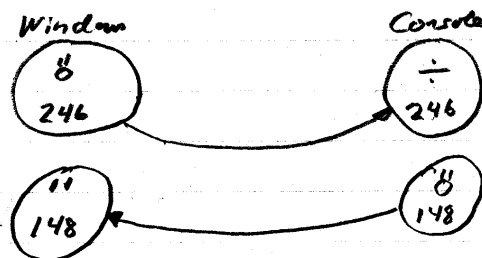
14

56

Varför ÷ på skärmen istället för ö ?

Beror på att ASCII-koden för tecken i

Windows och console(005) är olika.



(9)

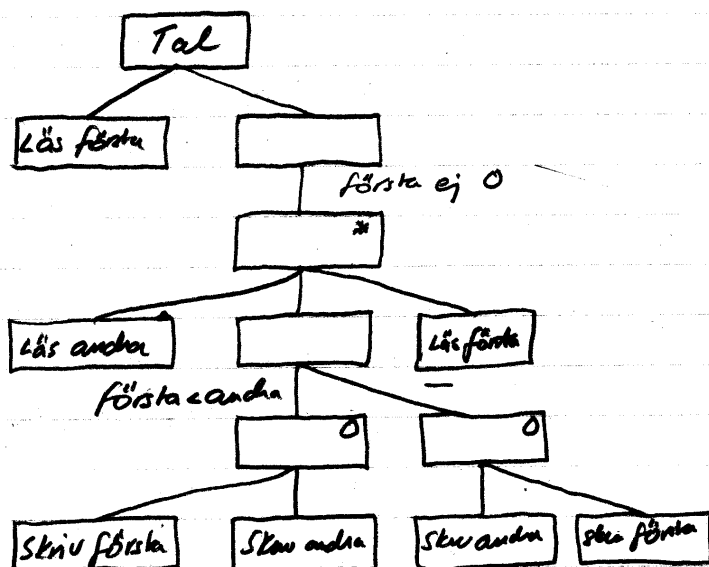
Hur får vi rätt tecken?

ALT1) $\text{printf}(\text{"Ge f\224rsta talet (avslut 0): "});$
148 oktalt

ALT2) $\text{printf}(\text{"Ge f\%c rsta talet (avslut 0): "}, 148);$

OBS! Skriver ut tecknet som
har ASCII-koden 148!

Documentation - för att dokumentera programmet
kan man rita struktendiagram.



10

(Ex) Skriv ett program som upprepat läser in ett tecken (avslut endast ENTER) och hur många gånger tecknet ska skrivas ut på skärmen.

```
void main()
{
    char tecken; // char som 'character'
    int antal, ggr;

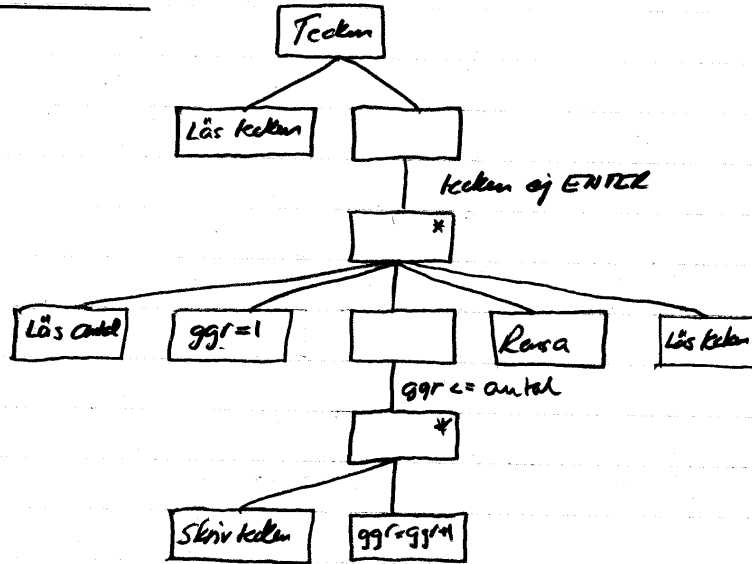
    printf("Ge ett tecken (avslut ENTER) : ");
    scanf("%c", &tecken); // OBS! %c

    while (tecken != '\n') // ENTER
    {
        printf("Ge antal gånger : ");
        scanf("%d", &antal);
        ggr = 1;
        while (ggr <= antal)
        {
            printf("%c", tecken);
            ggr = ggr + 1;
        }
        getchar(); // Rensar ett tecken från inbuffert
                // ALT) fflush(stdin);

        printf("\n Ge ett tecken (avslut ENTER) : ");
        scanf("%c", &tecken);
    }
}
```

(11)

Dokumentation



Hemuppgift En dator kan bara stiga på två saker 0 och 1, För man bara ha två symboler måste exempelvis tal kodas på ett annat sätt jämfört med vårt vanliga 10-system.

1 = 1 4 = 100 7 = 111
2 = 10 5 = 101 8 = 1000
3 = 11 6 = 110 |

skriv ett program som läser ett 8 bitars binärt tal och skriver ut motsvarande decimala.

Kör Ge ett 8 bitars binärt tal : 00000101
Talet är 5.

↑ ↑
bit nr = 7 bit nr = 0

Hint Läs binärkodens tecken för tecken och bilda tal
med $\text{tal} = \text{tal} + \text{pow}(2, \text{bit_nr})$

(12)