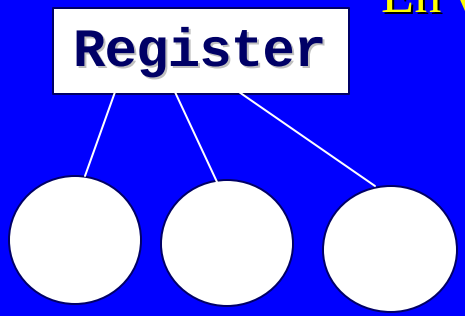
Skizz till en enkel databas

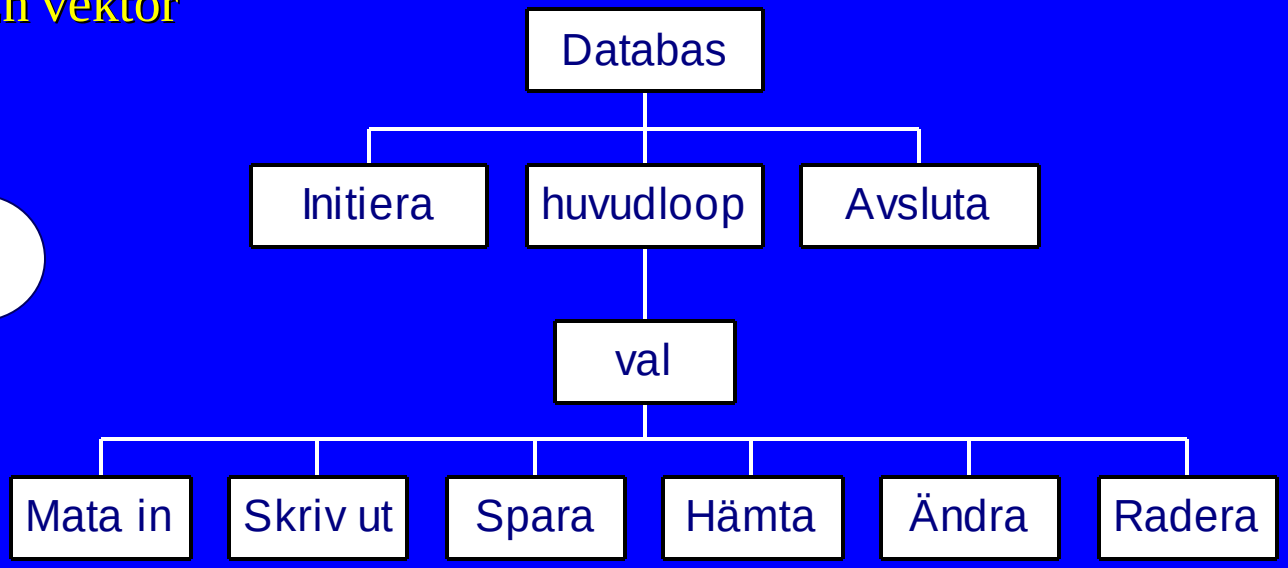
Data:

En vektor



Poster (struct)

Funktioner:



Enligt diskussion

Datastruktur

Vi behöver dataobjekt för att hålla samman poster med uppgifter som

- **`Namn`**
- **`Gatuadress`**
- **`Postnummer`**
- **`Ort`**
- **`Telefonnummer`**
- **`e-postadress`**

Lämpligen en egen datatyp struct

Datastruktur

```
// Förslag till datatyp för poster

struct Person{          // fungerar som egen datatyp
   char enamn  [20];
   char fnamn  [20];
   char gata   [20];
   char postnr [7];
   char ort    [15];
   char telefon[12];
   char e-post [20];
};
// Deklaration av ett dataobjekt
Person kompis;
```

Test av Person

```
Person lasInPost(){
   Person inpost;
   cout << "Skriv in data för en kompis!\n";
   cout << "Förnamn    :";
   cin.getline(inpost.fnamn, 20);
   // --------- osv...
   cout << "Ort        :";
   cin.getline(inpost.ort, 15);
   cout << "E-post     :";
   cin.getline(inpost.epost,20);
   return inpost;
}
```

Lokal post

Returneras

Struct och funktioner

En struct är en egendefinierad datatyp

Persontest.cpp

En struct kan skickas som argument

```
void skrivUt( Person p );
```

En struct kan returneras från en funktion

```
Person lasIn();
```

Data som lagrats i en struct transporteras via stacken

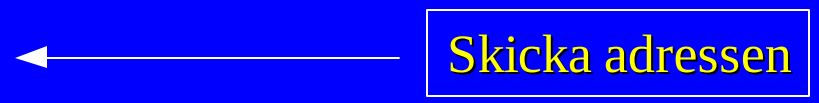
på samma sätt som enkla datatyper.

Struct och funktioner

Ofta lämpligt att skicka med en pekare till en struct

```
// Person deklareras här ovan :)
// Deklaration av funktionen fyllPa
void fyllPa( Person* p );
void main( void ){
  Person kotte;
  fyllPa( &kotte );
}
// Definition av funktionen fyllPa
void fyllPa( Person* p ){

}
```



Att komma åt element

”Vanlig” åtkomst med punktnotation

```
Person kotte;
cin >> kotte.fnamn;
```

Åtkomst via pekare

```
Person *pek = &kotte;
cin  >> (*pek).enamn;
cout << (*pek).fnamn;
```

Persontest2.cpp

Parentesen behövs för att . har högre prioritet än *

Förenklat skrivsätt för struct-typer

```
cin >> pek->fnamn;
```

Att komma åt element

```
// Förenklad notation för pekare till  struct

void fyllPa(Person* inpost){
   cout << "Skriv in data för en kompis!\n";
    cout << "Förnamn    :";
    cin.getline(inpost->fnamn, 20);
    cout << "Efternamn  :";
    cin.getline(inpost->enamn,20);
    cout << "Gatuadress :";
    cin.getline(inpost->gata, 20);
   // ----- > osv..
}
```

Persontest3.cpp

Flera poster

En databas med flera poster kan vara en array

```
const int MAX_ANTAL = 10000;
Person dbase[ MAX_ANTAL ];

// Mata in data från tangentbordet
mataIn (dbase);

// Skriv ut data till skärmen
skrivUt(dbase);

// osv
```

Flera poster

Inmatning av flera poster

```
mataIn(Person*  lista, int n){
  Person* pek = lista;
  for (int i = 0; i < n; i++){
       fyllPa( pek );
       pek++;
  }
}
```

Lokal pekare för stegning

Fyll på en i taget

Alla funktioner som behandlar poster fungerar likadant

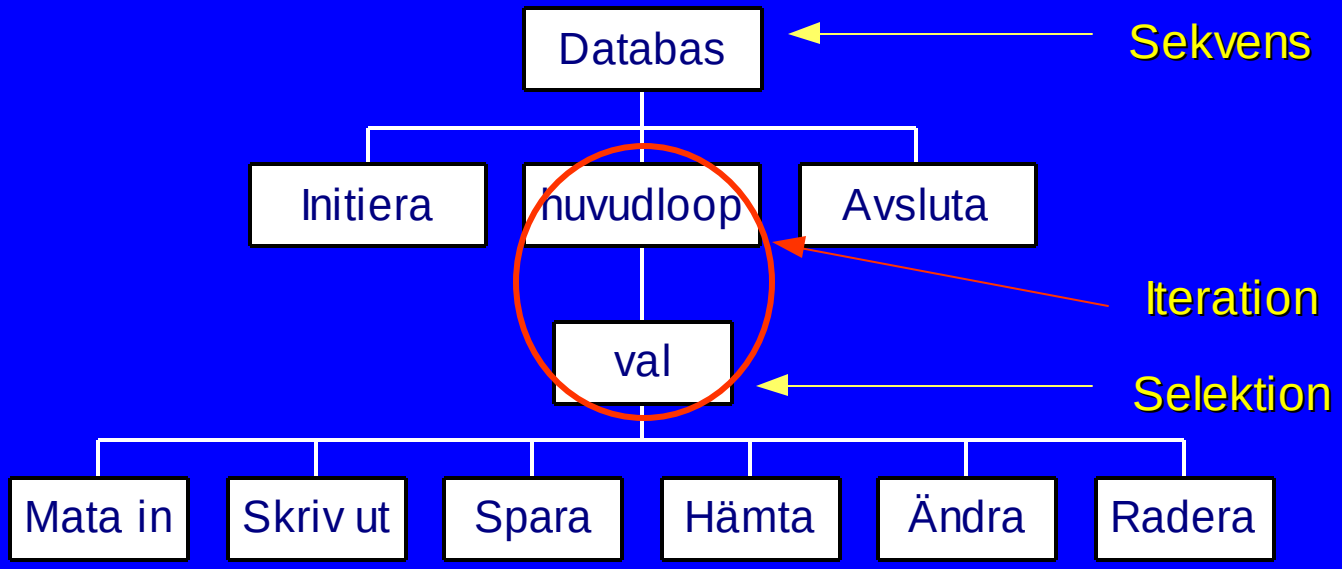
När ska man sluta?

Persontest4.cpp

Databasen - möjlig metod

Skapa datastruktur och alla funktioner

Fyll en funktion i taget med innehåll



Databasen - pseudokod

```
Databas
    initiera
    loopa [ do ... ]
        välj aktivitet
                utför vald aktivitet [ switch (val) ]
    så länge val < > ”sluta”
    avsluta programmet
Slut på databas
```

Databasen - ett skelett

Vi behöver

initiering - för variabler - berätta vad programmet gör
huvudloopen - ger valmöjlighet - händelsestyrning
avslutningen - sköter eventuell städning

Samt funktionerna

mataIn	case ’M’
skrivUt	case ’U’
spara	case ’S’
hamta	case ’H’
andra	case ’D’
radera	case ’R’
avsluta	case ’A’

Databasen - loopen

```
// Huvudloop
char val;
do { // minst en gång
     visaMeny();
     cin >> val;
     switch(val){
       case ’M’: mataIn();
       case ’U’: skrivUt();
       …
       case ’A’: // Avsluta programmmet
       default: cout << ”Valet finns inte”;
     }
  } while(val != ’A’);
```

Skelett.cpp

Händelsstyrning

En loop som körs minst en gång

```
do { //välj } while( <villkor> );
```

Inmatning från användaren avgör vilken kod som körs

Villkoret måste ändras i loopen

```
cin >> val; val = toupper(val);
```

Urval av kod görs med **switch**-sats

```
switch(val){
  case 'S': spara();break;
  case ' ': // något annat
```

Felaktig inmatning måste tas omhand i loopen

```
default: cout << "Felaktigt val. Försök igen!"
```

Databasen - inmatning

En tvåstegsprocess

```
// Inmatning av ett antal poster

void mataIn(Person* l, int n){
   Person* pek = l;
    for (int i = 0; i < n; i++){
        fyllPa(pek);
        pek++;
    }
}
```

Databasen - inmatning

Inmatning till en av posterna

```
void fyllPa(Person* inpost){
    cout << "\nSkriv in data för en kompis!\n";
    cout << "Förnamn    :";
    cin.getline(inpost->fnamn, 20);
    cout << "Efternamn  :";
    cin.getline(inpost->enamn,20);
    cout << "E-post     :";
    cin.getline(inpost->epost,20);
}
```