Lektion 3 – Vektorer och strängar

Vektorer

En vektor används för att spara flera olika värden. Om du exempelvis vill spara längden hos alla i klassen borde du använda dig av en vektor. För att skapa en vektor som sparar heltal skriver du

```
vector <int> langder;
```

Om man inte skriver mer kommer vektorn inte innehålla några tal. För att mata in exempelvis 180 i vektorn skriver man `langder.push_back(180)`.

För att senare få tillgång till element i skriver man `langder[i]`, det kommer då fungera som en vanlig variabel. Man kan alltså ändra element i till 163 genom att skriva `langder[i] = 163;`

Låt oss skriva koden för ett program som läser in längden på n stycken personer och sparar värdena i en vektor.

```
// skapar variablerna n och langd
int n, langd;

// skapar vektorn som ska spara klassand langder
vector <int> langder;

// matar in hur manga det ar i klassen i n
cin >> n;

// skapar en for-loop som upprepas n ganger
for(int i = 0; i < n; i++){
    // matar in langden pa en elev i langd
    cin >> langd;
    // matar in langden pa eleven i vektorn
    langder.push_back(langd);
}
```

Strängar och bokstäver

Vi har tidigare nämnt att man kan spara text i strängar och nu ska vi gå in lite djupare i hur de fungerar. Man kan se en sträng som en vektor som innehåller variabler av typen `char`. En `char` är bara en bokstav, och för att skapa en `char` som heter `min_bokstav` skriver vi

```
char min_bokstav;
```

För att sedan spara exempelvis `c` i `min_bokstav` skriver vi `min_bokstav = 'c'`. Notera att man skriver `'` istället för `"`.

Som exempel skapar vi en sträng som sparar ett namn.

```
string namn = "Teodor";
```

För att ändra i varibeln `namn` gör man likadant som en vektor och skriver `namn[0]` för att komma åt den första bokstaven. Exempelvis kommer `cout << namn[0] << endl;` skriva ut `T`. Jag kan också ändra namnet genom att skriva

```
namn[0] = 'F';
cout << namn << endl;
```

Programmet kommer nu skriva ut `Feodor` istället för `Teodor`.