6 Løkker

Læs om while-løkker i kapitel 3.3 og om for-løkker i kapitel 3.4. En løkke er en sekvens, som skal gentages et antal gange. Kender vi antallet af iterationer, benytter vi en for-løkke; kender vi ikke antallet af iterationer, benytter vi en while-løkke.

- Vask disse 5 tallerkener op. For-løkke
- Hvor længe skal jeg lede efter min nøgle? Til den er fundet. While løkke.

6.1 While-Løkke

While loop bruger vi, når vi ikke ved, hvor mange iterationer vi skal bruge. En iteration er hver gang, løkken gentager sig selv.

Vi kan oversætte det engelske ord while med så længe.

```
boolean found=false;

while(!found){
        // set found til true for at stoppe

}
```

Figur 8: While løkke

Hvis vi opretter en variabel af typen boolean og navngiver den "found"og initierer den med værdien falsk, kan vi benytte os af en negation (!)(ikke) og bruge betingelsen while(!found). Det læses "while not found". Hvilket giver god mening, at vi skal blive ved med at lede efter bilnøglen, til vi finder den. Tænker vi over det, skal while-løkken have en sand betingelse, men værdien for found er falsk. Men !found gør den sand. I det øjeblik vi finder, hvad vi leder efter, sætter vi found til at være sand.

6.2 For-løkke

For-løkker bruges, når vi kender antallet af iterationer. For eksempel hvis vi skal løbe alle bogstaver igennem i en streng eller objekter i et array. Fordelen ved en for-løkke er, at der er indbygget en tæller, som vi kan bruge.

```
for (int i=0;i<str.lenght();i=i+1){
        // kode som gentages
}
```

Figur 9: For-løkke

En for-løkke består af tre dele. Del 1 er en deklarering og initiering af vores index/tæller. Vi kalder index for i. Har vi brug for flere indlejrede løkker, følger vi alfabetet og kalder den næste j så k etc. Del 2 er den betingelse som skal være opfyldt for at løkken bliver udført. Betingelsen knytter sig typisk til vores index i og længden af den streng eller array vi benytter. For eksempel: $i <= 10$ eller $i < str.length()$. Del tre beskriver den handling som skal ske med i efter hver iteration. Vi kan tælle i op med 1; i++. Eller trække en fra med En for-løkke består af tre dele. Del 1 er en deklarering og initiering af vores index/tæller. Vi kalder index for i. Har vi brug for flere indlejrede løkker, følger vi alfabetet og kalder den næste j, så k osv. Del 2 er den betingelse, som skal være opfyldt for at løkken bliver udført. Betingelsen knytter sig typisk til vores index i og længden af den streng eller array, vi benytter. For eksempel: $i <= 10$ eller $i < str.length()$. Del 3 beskriver den handling, som skal ske med i efter hver iteration.$i--$.

6.3 Enhanced loop

Enhanced loop benyttes til at iterere gennem en liste af objekter og udføre en instruktion for hvert element i listen.

```
for (Particle part : particles) {
  part.display();
}
```

Figur 10: Enhanced-løkke

Vi har talt om, at en string har et index. Med en løkke kan vi gennemløbe alle positioner af en streng og se på det enkelte bogstav på den i'ne-plads.

Slå op i dokumentationen for Processing og læs om string(). Nederst er der en række metoder, som kan benyttes sammen med en string. Når en datatype starter med stort bogstav, vil der altid være metoder, I kan bruge. Man bruger metoderne sammen med variabelnavnet. F.eks. hvis str er defineret som datatype String, så kan man skrive: str.charAt(i);

6.4 Opgave

Lav henholdsvis en for-løkke og en while-løkke, som udskriver de 5 første bogstaver i sætningen "Hej med dig!". Tip: brug funktionen charAt() sammen med din tæller.

6.5 Opgave

Lav henholdsvis en for-løkke og en while-løkke, som skal finde alle e'er i sætningen: "Dette er en sætning, som indeholder mange e'er. Men hvor mange er

der?"Løkken skal udskrive alle e'er og udskrive, hvor mange e'er som er fundet. Tilsidst udskrives længden af ovenstående sætning.

6.6 Opgave

Du skal lave et while-loop, som sammenligner alle bogstaver i en streng med et 'Z'. Når betingelsen er sand, skal løkken stoppe. Udskriv "Fundet"til konsollen sammen med index for bogstavet 'Z'. Brug denne streng "I Afrika lever der mange dyr på savannen, et af dem er zebraen. Zebraen er en flok dyr."Tip: du kan med fordel bruge str.toLowerCase(), hvis din streng hedder str, ellers skal du lede efter både 'z' og 'Z'. For at sammenligne et tegn kan du bruge betingelsen: str.charAt(i)=='Z'.

6.7 Opgave

Denne opgave ligner den tidligere, men der er forskel. Du skal lave et for-loop, som tæller alle 'Z' eller 'z' i en streng. Udskriv "antal z'er: "til konsollen sammen med antallet af z'er. Brug denne streng: "I Afrika lever der mange dyr på savannen, et af dem er zebraen. Zebraen er et flokdyr."

6.8 Opgave

Omskriv nu de to opgaver, så du bruger for-løkke i while-opgaven og vice versa.

7 Funktioner

Læs om funktioner i kapitel 3.5.
Vi skal arbejde med funktioner. Du kender dem fra matematikken $f(x)$, hvor en funktion beregner/returnerer en værdi. Vi bruger funktioner, når vi skal udføre den samme sekvens flere gange med forskellige variabler. Funktioner i Java består af 4 dele.

Navn: Funktioner skal navngives med et ikke-reserveret ord (ord som er brugt i forvejen), men hvor navnet er sigende for funktionens funktion. I mit eksempel er navnet "minFunktion". Vi bruger navnet, når vi skal bruge funktionen.

Returdatatype: Funktioner skal deklareres efter returdatatype. Funktionen kunne returnere en int, float, string eller char, men altid kun én ting. Hvis funktionen ikke returnerer noget, skal den defineres som void.

Parameter: Funktionen kan modtage en lang række forskellige parametre. En funktions parameter skal angives i parenteser sammen med, at vi deklarerer datatypen. Man kan deklarere mange parametre til sin funktion. Parametre er kun gyldige lokalt. Det vil sige, at du ikke kan bruge en variabel, du har deklareret i andre funktioner, uden at skulle deklarere dem igen.

Kode: En funktionskode skal skrives imellem de to Tuborg-parenteser .

```
float pris=20.0f;

void setup(){
        size(400,400);
        pris=beregnMoms(pris);
        udskrivPris(pris);
}

//funktion som beregner moms
float beregnMoms(float pris){
        return pris*1.25;
}
// funktion som udskriver pris
void udskrivPris(float pris){
        println("Pris incl moms: "+pris);
}
```

Figur 11: Funktion

7.1 Opgave

Skriv et program, hvor dit canvas er 800x800. Opret en funktion, som tegner en cirkel i midten af dit canvas. Cirklen skal være rød. Navngiv din funktion, så den fortæller, hvad funktionen gør. Opret en funktion, som tegner en firkant. Firkanten skal være blå. Navngiv din funktion, så den fortæller, hvad funktionen gør.

7.2 Opgave

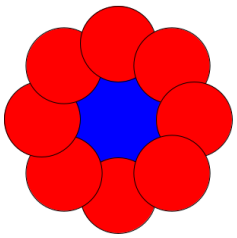
Tilpas din funktion, så den tegner 8 cirkler. Brug et for-loop, og brug dit index til at gange din x-koordinat i cirklen, således at din cirkel ikke bliver tegnet det samme sted.

```
for (int i = 0; i < 10; i++){
// flyt cirkel med brug af i.
   circle(0+i*20,0,20);
}
```

Figur 12: For-løkke

7.3 Opgave

Lave en blomst af cirkler. Placer en firkant i midten, og lav kronbladene af 8 cirkler. Du kan måske bruge funktionerne pushMatrix(), popMatrix(), translate() og rotate()?



Figur 13: Blomst