# Introduktion til programmering

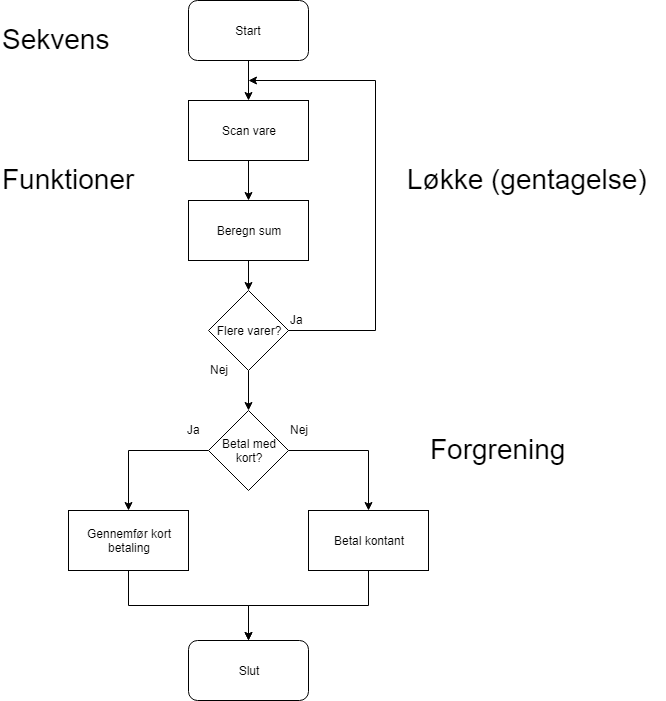
Udarbejdet af Lars Skjærbæk, UCH

Et program er en instruktion til computeren, som fortæller den hvad den skal gøre. Programmet består af en række instruktioner eller programtrin, som kan oversættes til koder, som computeren kan forstå og udføre.

Programmer skrives normalt i kodesprog som f.eks. Java, C eller Python (links til eksempler), hvor der er meget stringente krav til den måde, man skriver koden på. For at gøre det lettere at komme i gang med at programmere, er der lavet nogle værktøjer, hvor du grafisk kan sammensætte programmer ved hjælp af nogle blokke. En fælles betegnelse for disse programmer er blokprogrammering, hvilket bl.a. benyttes i Scratch og på Code.org.

## Grundbegreber i programmering

For at kunne programmere, er der nogle begreber vi skal kende. Begreberne er vist i nedenstående flowchart, som viser købeprocessen ved kassen i et supermarked.



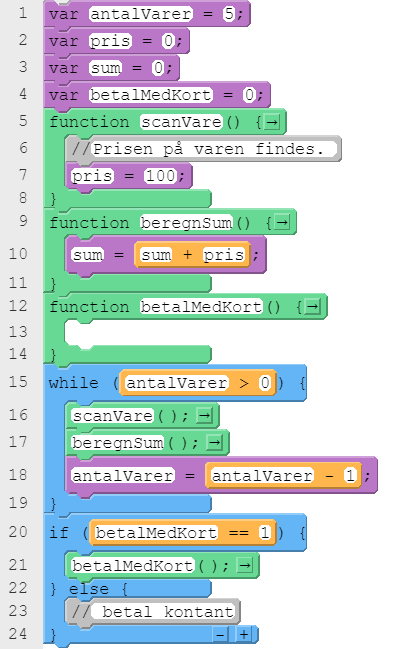
En sekvens er et eller flere programtrin, som udføres efter hinanden.

En funktion udfører en opgave (f.eks. gennemfør kort betaling). Man behøver ikke vide, hvordan den gør det, med mindre man selv skal udvikle funktionen. Man kan bare bruge den. Derved kan man abstrahere fra en masse detaljer.

En løkke er når et eller flere programtrin gentages. I det her tilfælde spørges der på, om der er flere varer, der skal scannes. Hvis ja, gentager vi ”Scan vare” ellers går vi videre til betalingen.

En forgrening er, når vi spørger på, om en betingelse er opfyldt. I det her tilfælde spørger vi på, om der skal betales med kort. Hvis ja, så gennemføres kortbetalingen. Ellers betales der kontant.

## Blokprogrammering

Et blokprogram til ovenstående eksempel kan se ud som vist i figuren til højre.

Først definerer man variable. En variabel er en ”skuffe” som kan indeholde en værdi. Variable kan være af forskellige typer (tal, tekst, bit). I blokprogrammering finder programmet selv ud af, hvilke typer man arbejder med. I det her tilfælde er alle variable af typen tal.

I blokprogrammering ligger der en masse funktioner, som gør det nemt at skrive et program. En funktion kan f.eks. hente input fra brugerinterfacet, eller skrive noget ud på skærmen. Man kan også lave sine egne funktioner (de grønne blokke). I eksemplet her, har vi oprettet funktionerne: scanVare(), beregnSum() og betalMedKort(). For at gøre eksemplet simpelt, sætter vi bare en fast pris på varen på 100 kr., og funktionen betalMedKort() har vi indtil videre abstraheret fra. Funktionerne kaldes længere nede i programmet.

En løkke (gentagelse) laves eksempelvis ved brug af en ”while” blok (se første blå blok). Hvis betingelsen er opfyldt, her antalVarer > 0, så udføres de programtrin, som er inden i løkken. I løkken tælles antalVarer ned, og når den når 0, så stopper løkken og programmet fortsættes.

En forgrening laves med en if-blok (sidste blå blok). Hvis betalMedKort==1, så kaldes funktionen betalMedKort. Ellers betales der kontant. Når der er to lighedstegn, er det fordi man tester på, om BetalMedKort er lig med 1. Er der kun et lighedstegn, sætter man betalMedKort til 1.

## Øvelse: Lær at programmere på code.org.

Blok programmering kan læres på code.org, som er en ”non profit” organisation, der ønsker at øge tilgangen til computer videnskab i skolerne. De har udviklet nogle online kurser, som i dag benyttes af millioner af elever over alt i verden, og som derfor er meget velafprøvede. Kurserne er rette mod børn i forskellige aldre, men de er også velegnede til større elever (og lærere) som ikke har kode erfaring.

Her på sitet finder du en video tutorial (Link: Start kodning), som forklarer hvorledes du kommer i gang med code.org. Jo flere kurser du gennemfører, desto mere trænet bliver du i computationel tænkning og i kodning.

Det næste er, at du skal have eleverne til at gennemføre kurserne. Hertil er der udviklet nogle smarte værktøjer, som gør, at du kan følge med i elevernes progression. Også her er der en video tutorial (Link: Opfølgningsværktøjer), som kan hjælpe dig i gang.