Contents

1 Sekventiel/procedural -programmering

Denne opgave handler om sekventiel og procedural programmering. Sekventiel programmering, betyder at instruktionernes rækkefølge ikke er ligegyldig. Procedural programmering betyder at vi kan opdele vores program i forskellige procedurer eller funktioner og bare kalde proceduren når vi har brug for den. Det ville være smart med en procedure som kan beregne en vares pris med moms.

Det vigtige i denne opgave er, at du skal lære at bruge dokumentationen til processing/Java.

Herudover vil du blive introduceret til rutediagrammer. Et værktøj som skal hjælpe dig med at strukturere dine programmer.

Der er altså hele tre ting som kræver din opmærksomhed.

1.1 Opgave

Du skal lave en kopi af min tegnig: opg1-hoejhat.pdf. Det primære fokus i denne opgave er, at bruge dokumentationen i processing. Der findes i processing en lang række forud definerede procedurer/funktioner. Vi kan se at det er en funktion fordi er altid er () bagefter. For eksempel size(); Size er en funktion som kræver to parameter, brede og højde. size(400,600); Men hvordan finder man ud af hvor mange parameter og hvilke man kan bruge til en funktion? Det gør man ved at kigge i dokumentationen til processing. Du finder alle funktioner i processings reference guide: processing.org/reference.

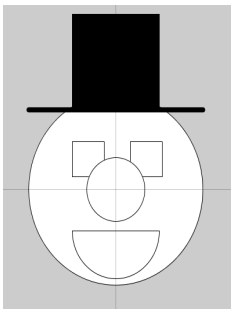

Figure 1: Højhat

Du kan bruge disse otte instruktioner for at kunne lave tegningen.

- size();
- line();
- strokeWeight();

- rect();
- square();
- circle();
- arc();
- fill();

Brug Processings dokumentation: processing.org/reference, for at finde ud af hvilke parameter de forskellige funktioner skal have.

- canvas (vindue som processing åbner) kan have størrelsen $400, 600$
- strokeWeight() er tykkelsen på stregen.
- fill() udfylder figuren med en RGB-farve.

Husk at koordinaterne $0, 0$ er øverste venstrehjørne (normalt vil det være nederste venstre) og er efter princippet: "hen ad vejen, ned til stegen". X,Y.

Brug rutediagrammet i figur 2 for at skrive koden:

Hvad sker der, hvis du bytter om på rækkefølgen? Altså hvis du starter med øjne,næse og mund og så tegner ansigtet.

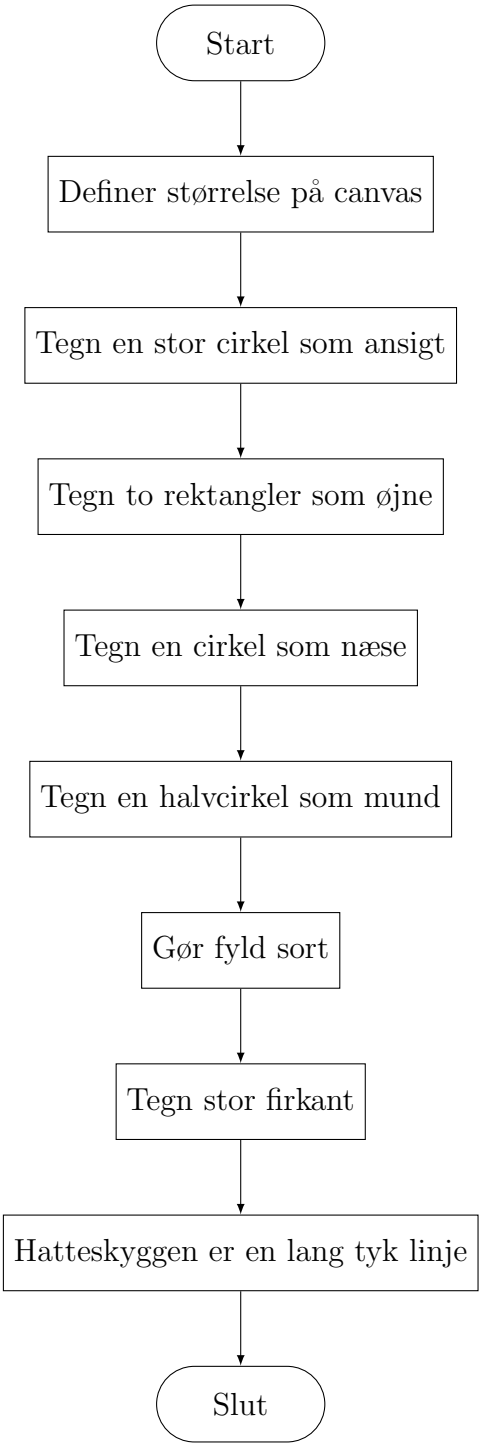

Figure 2: Rutediagram

1.2 Opgave

Du har nu en fornemmelse af hvad sekventiel programmering er. Men en af styrkerne i Java er, at det understøtter produceral programmering. Hver procedure programmeres sekventielt. I Proccesing skal der altid være to procedurer. Setup og draw. Setup bruger vi til f.eks. at sætte størrelse på canvas eller andre initieringer. Draw er hoveddelen. Proceduren (vi kalder det også en funktion) looper 60 gange i sekundet. Vi kan selv definere alle de procedurer som vi har brug for.

Tilføj nu følgende linjer til dit program og flyt alt dit kode ned i proceduren/funktionen hoved.

```
void setup() {
                    size(480, 120);
}
void draw() {
                    head();
}
void head(){
                    // Din kode skal stå her! I mellem de to tuborg paranteser.
}
```

1.3 Opgave

Del dit program yderligere op. Således at én funktion tegner hatten. En anden tegner øjne, en tegner munden og den sidste tegner ansigtet. Kald funktionerne for: void hat(){} ,void eyes(){} ,void mouth(){}, void face(){}, husk at fjerne funktionen head(), når du er færdig.

1.4 Opgave

Gør nu dit canvas så stort at der er plads til to hoveder ved siden af hinanden. Tilføj to parametere til dine funktioner, en float x og en float y koordinat, udfra hvilke du kan tegne alle dine elementer af ansigtet. Ret dine linjer til så de tager udgangspunk i dine x og y koordinater. Er du rigtig god, kaler du dine funktioner med de samme værdier!

1.5 Opgave

Tegn figuren færdig med krop, arme og ben i hver deres funktion.

2 Primitiv Animation

Det vigtige i denne opgave er, at du skal lære at bruge dokumentationen til processing/Java. Herudover skal du bruge rutediagram til at strukturere din kode. Tænk på animation som stop motion teknik.

Vi skal bruge om nogle nye instruktioner

- frameRate()
- frameCount()
- noLoop()
- rotate()
- translate()
- popMatrix()
- pullMatrix()

Brug Processings dokumentation: processing.org/reference, for at finde ud af hvad de forskellige funktioner gør og hvilke parameter de forskellige funktioner skal have.

I sidste kapitel introducerede jeg proceduralprogrammering. Husk at processing altid skal have to funktioner: void setup() void draw() Herefter kan man selv definere sine egne funktioner.

2.1 Opgave

Lav et program, hvor et hjul med minimum tre eger ruller over skærmen. Start med at lave et rute diagram.

2.2 Opgave

Udvid programmet så når hjulet forsinder dukker det op på den anden side igen.

2.3 Opgave

Lav programmet om så når hjulet rammer væggen, at det stopper og ruller tilbage hvor det kom fra.

2.4 Opgave

Lav et program som animerer en mand som kan gå. Start med at lave et rute diagram.

Du kan finde inspiration i mine to eksempler.

3 Lektion 3 uge 37

Du kender nu til følgende datatyper og instruktioner:

Datatyper:

- float: kommatal.
- double: mere præcis end float.
- char: en enkelt karakter. Char er en unsigned byte og kan derfor ikke indeholde minustal.
- int: heltal.
- long: 2 gange så stor som en integer.
- boolean: sand eller falsk.

Instruktioner:

- size(); // sætter størrelsen for canvas
- line(); // tegner en linje
- stroke(); // farven på en streg
- strokeWeight(); // tykkelse af streg
- rect(); // tegner en rektangel
- circle(); // Tegner en cirkel
- arc(); // Tegner en bue
- fill(); // udfylder en figur med farve. [1]

I denne opgave skal du bruge datatypen integer. Integers er heltal som f.eks: 0,1,2,3,4,5... i java deklarerer vi variablen x af datatypen Integer som:

```
int x;
```

Eksempel på en deklaration og initiering i samme linje:

[1] Find rgb farver her: www.rapidtables.com/web/color/RGB_Color.html

```
int x = 0;
```

Vi skal også 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 navenet "minFunktion". Vi bruger navnet når vi skal bruge funktionen.
- Returdatatype: Funtioner 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 parameter. En funktions paramenter skal angives i parantesen efternavenet samtidig med at vi deklarerer datatypen. Man kan deklarere mange parametere til sin funktion. Parameter er kun gyldige lokalt. Det vil sige at du kan ikke bruge en variabel du har deklareret i andre funktioner uden at skulle deklarere dem igen.
- Kode: En funktions kode skal skrives imellem de to tuborgparanteser .

Hvis du vil bruge funktioner, så husk void setup();/ og void draw();

3.1 Opgave

3.1.1

Se programmet fourTimesAAlien (hentes på GITHUB).

1. Find funktionen frameFunction. På hvilken linje (nummer), deklarerers frameFunction() ?
2. Fra hvilke linjenumre, bliver funktionen kaldt?
3. Med hvor mange parameter kalder jeg funktionen?

3.1.2

Opret nu dine egne funktioner som laver:

1. Hoved med øjne
2. Kroppen med ben
3. Skirver et nyt navn
4. Kald dine tre nye funktioner fra draw() og se om programmet stadigvæk tegner en alien i det øverste felt til venstre.
5. Opret en ny funktion som kalder hoved og krop og ret draw() til. Din funktion skal tegne en alien i alle firkanter. Du skal tilpasse parameterne og du finder nok ud af at det har noget med x,y pos at gøre :)

3.1.3 Opgave

Du skal lave et kasseapperat. Apperatet skal have nogle forskellige funktioner som returnerer forskellige værdier. Når en funktion returnerer en værdi skal vi have en beholder som kan indeholde den værdi som kommer retur. F.eks. en moms funktion returnerer en float.

//deklaration af variablen float belobMedMoms=0; // kald funktionen og hold beholderen beloebMedMoms klar til at modtage retur værdien fra funktionen. belobMedMoms = beregMoms(belobUdenMoms);

Hent programmet kasseapparat fra Github

1. Se på funktion som beregner moms. Den modtager en integer værdig som parameter og returnerer en float(kommatal). Du beregner momsen ved at gange med 1,25. Forklar alle linjer i programmet.
2. Lav en ny funktion som trækker moms fra. Den modtager en float værdi som parameter og returnerer en float. Man trækker momsen fra ved at gange med 0,8.
3. Lav en ny funktion som kan udskrive en bon.

   Tak for dit køb Jens, Du har købt for 125 kroner.

   Beløb uden moms: 100,- Beløb med moms: 125,- Momsbeløbet udgør: 25,-
4. Slå day() op i dokumentationen og tilføj følgende linje til din bon Dato: 16/9 2020 kl 13:00

   ingen mellemrum imellem talne :)