1 Fabrikken

Vi skal lave en fabrik som producerer varer i form af forskellige datatyper.

1.1 Bit-mining

Efter denne opgave ved du noget om:

- Funktioner og deres retur datatyper
- Den primitive datatype boolean
- Global og lokal variabler
- if else statement

En fabrik skal bruge råstoffer som jern, kul, plastik eller korn som man kan forarbejde til produkter. I programmering er råstoffet bits = 0 eller 1. Lav en funktion bitmining() som laver én bit og returnerer værediен sand eller falsk.

1.1.1 Psudokode

Nedbryd problemet til de mindst mulige enheder. Idéen er at lave en funktion som returnerer en boolsk værdi. boolean bitMine(){}
Processing har en indbygget tilfældighedsgenerator som returnerer et tilfældigt kommatal. Kommandoen hedder: random() - slå funktionen op i dokumentationen til processing så du forstår hvordan den virker.

- Opret en funktion bitmine som kan returnere en boolean
- Generer en tilfældig værdi mellen 0 og 2
- Hvis værdien er mindre en 1 returner falsk
- Ellers returner sand

Din funktion kan du sætte ind i denne skabelon: se figur 1 Det kan være svært at se hvad programmet udskriver.

- Gør teksten stor - textSize()
- Gør teksten sort - fill()

Linjerne skal du tilføje funktionen: void setup() under linjen: size(800,800);

```
void setup(){
        size(800,800);
}
void draw(){
        background(205);
        // hvis bitmine returnerer true er det 1
        //ellers er det nul
        if(bitmine()){
                text(1,50,50);
        }
        else{
                text(0,50,50);
        }
}
boolean bitmine(){
        // skriv din funktion her
}
```

Figur 1: Bitmine

1.2 Byte produktion

Efter denne opgave ved du noget om:

- Funktioner og deres retur datatyper
- Den primitive datatype byte
- Det binære talsystem
- Global og lokal variabler
- for løkke

Hurtigt står det klart at vores produkt ikke kan klare sig på det globale marked og at vi derfor er tvunget til at øge vores værdikæde. En bit er ikke noget værd (max 1) før vi får 8 bits samlet til en byte. Derfor vil vi gerne udvide vores produktion med en afdeling som kan samle 8 bits til en byte. byte makeByte(){} funktion.

1.2.1 Psudokode

Min idé til denne funktion er at bruge det binære talsystem sammen med et forloop. I det binære talsystem angiver den første position (fra højre) alle 1'ere (2^0) . Den anden position alle 2'erer (2^1). Den tredje position alle 4'ere (2^2)og så videre. se kapitel 1 om det binære talsystem. Vi kan derfor bruge et for loop til at beregne: 2^i.

```
byte makeByte(){
        byte b; //lokal variabel til resultatet
        for (int i=0;i<8;i++){
                //Beregn 2 i i potens og plus til b;
        }
        return b; //retuner resultatet
}
```

Figur 2: makeByte

1.3 Char produktion

Efter denne opgave ved du noget om:

- Funktioner og deres retur datatyper
- Den primitive datatype char
- Global og lokal variabler
- While løkke

Vores virksomhed ekspanderer og hurtig står det klart at vi har ramt rigtig i markedet med vores produkt. Vi har dog mulighed for at ramme et smalt segment i markedet hvis vi tilpasser vores produktion med et produkt som kan repræsentere en karakter, den primitive datatype char.

1.3.1 Psudokode

Problemet er at datatypen byte er signed, og at datatypen char er unsigned. (se kapitel 1 om binære talsystem). En byte kan rumme værdierne -128 - 127 og en char værdierne 0-255. Vi kan derfor ikke overføre en byte til et tal direkte, MEN! ASCII tabellen viser at A=65 og Z=90 og a=97 og z=122.

Værdierne er ikke negative og vi kan derfor type caste værdierne til en char med kommandoen: char(b).

- Opret en funktion som kan retunere en char og modtage en byte b
- deklarer en variabel b af typen byte og initier den med værdien fra makeByte()
- Retuner værdien af char(b)

1.4 String produktion

Efter denne opgave ved du noget om:

> Den ikke primitive datatype String
>
> funktioner knyttet til ikke primitive datatyper

Patentet på fabrikkens karakterer er ved at udløbe og derfor står vi overfor at miste en stor del af vores markedsandel, hvis ikke vi produktudvikler. Innovations afdelingen har været på overarbejde og det er lykkedes dem at finde på et nyt produkt, en sætning. En sætning svarer til den ikke primitive datatype String som indeholder en række af karakterer.
Det er et krav at første bogstav ordet er et stort bogstav og resten små. En krølle på produktet er at længden af sætningen skal være variabel og må ikke være mindre en 3karakterer.

1.4.1 Psudokode

En String er en række af karakterer. Så løsningen er, at oprette en variabel str af typen string og tilføje én karakter af gangen.

- Opret en funktion som retunerer en String og modtager en int strLength
- deklarer en variabel str af type String og initier den med værdien
- kald funktionen makeChar() strLength-gange med et for-loop og føj værdien til str
- Retuner str

1.5 Integer produktion

Produktionen af bits, bytes, char og strings kører nu perfekt. Salgsafdelingen fortæller nu om en øget efterspørgsel af Integer værdier. Den maksimale værdi for en integer er $2^{31} - 1$ og kan produceres af 4 bytes.
Lav en funktion som producerer integer værdier.

1.6 Forbedringer

Sammenlign dit ord med ord i denne ordsamling:
https://github.com/dwyl/english-words
Hvis fabrikken genererer et ord som er længere end 2 karakterer skal der udskrives "Tillykke"til skærmen. Lav en global tæller som holder styr på hvor mange gange det er sket. Brug evt. funktionen findWord() som du finder her 3.
Brug en arrayliste til at indeholde alle fundne ord. Udskriv alle ord til canvas. Hvem finder det længste ord?

```
String[] lines;

void setup(){
        size(800,800);
        lines = loadStrings("words.txt");
}
void draw(){

}

boolean findWord(String str){
for (int i = 0 ; i < lines.length; i++) {
  if(lines[i].euquals(str)){}
}

}
```

Figur 3: findWord

1.6.1 Visualisering af produktionen

Implementer en visualisering af hvordan produktionen foregår og hvordan de forsellige afdeliger arbejder sammen. Implementer visualiseringen i dit

program.