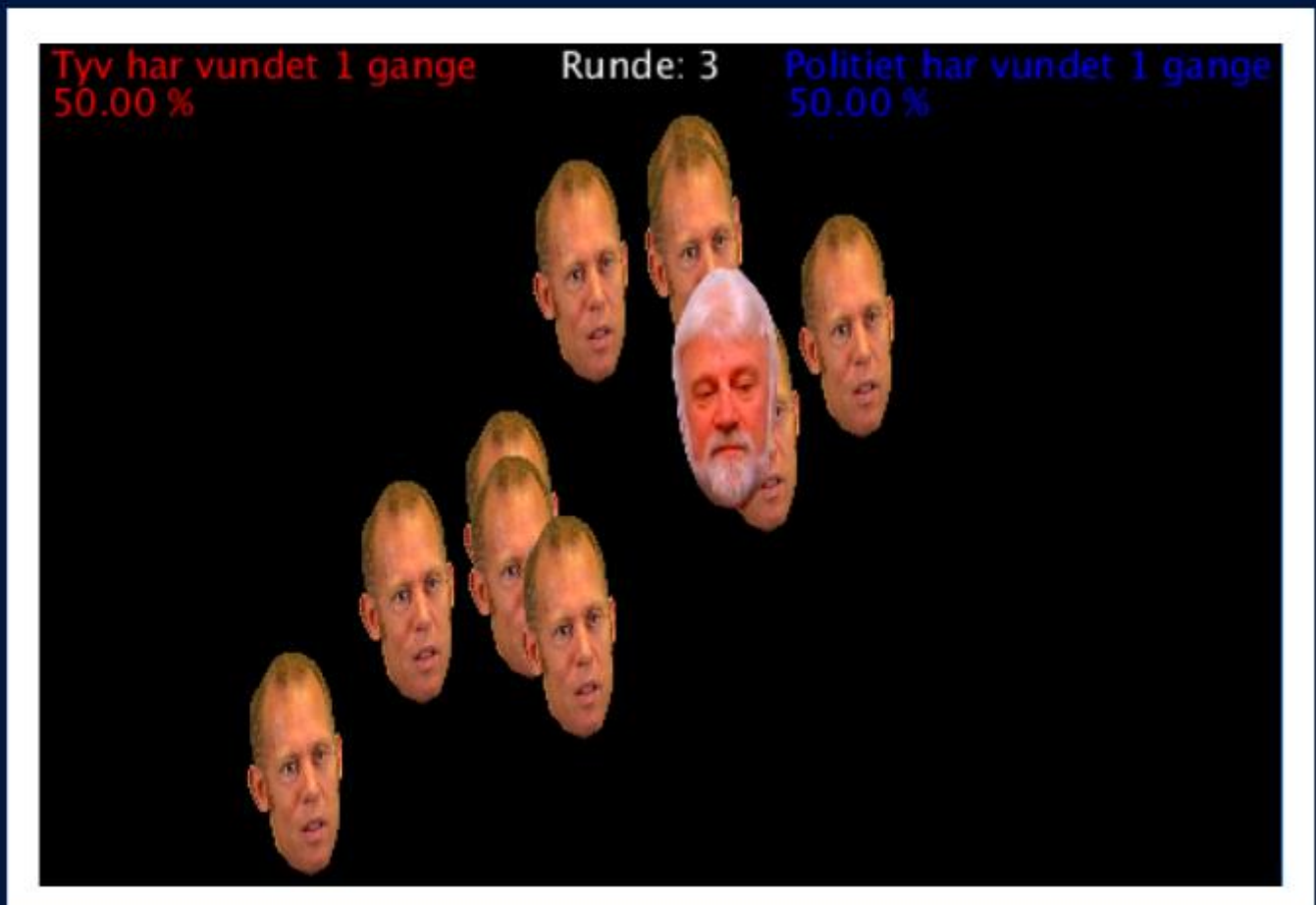


Cops and robbers



Max Hansen

*Odense Tekniske Gymnasium
Programmering B*

Indholdsfortegnelse

| | |
|------------------------------------------|----|
| Indledning..... | 3 |
| Kravspecifikation | 3 |
| Krav 1 | 3 |
| Krav 2 | 3 |
| Krav 3 | 3 |
| Kravanalyse..... | 3 |
| Krav 1 | 3 |
| Krav 2 | 4 |
| Krav 3 | 4 |
| Gennemgang af programmet | 6 |
| Opbygning af programmet | 6 |
| Flowdiagram | 6 |
| Klasser som der er brugt i spillet | 7 |
| Biblioteker som der er brugt | 7 |
| Test af programmet..... | 8 |
| Hvordan fungerer det? | 8 |
| Overholdes kravene? | 8 |
| Beskrivelse af arbejdsprocessen..... | 8 |
| Bilag | 9 |
| Bilag 1 Cops and robbers | 9 |
| Bilag 2 CopLib | 13 |
| Bilag 3 RobberLib..... | 14 |

Indledning

Cops and Robbers er et meget gammelt spil. Spillet har været så basalt at det er noget som man ofte har leget da man var et barn i frikvartererne i skoletiden. Spillet er både sjovt men også meget tilfældigt, da det hele afhænger af hvordan røveren og hvordan politiet bevæger sig i forhold til hinanden. Det vil altså sige, hvis røveren ikke tænker sig om, så bliver røveren fanget. Hvis det derimod er politiet som ikke tænker sig om, så slipper røveren væk. Jeg har derfor valgt at lave min version af spillet, hvor det er Morten Spiegelhauer som jager Rudy Frederiksen, da det er Rudy som er den onde person.

Kravspecifikation

For at kunne lave et spil, så kræver det at man sætter sig nogle krav for hvad der skal være med i spillet, samtidigt med at man skal sætte nogle krav for hvordan spillet skal fungere i praksis, så man har noget at gå efter, når man så endelig går i gang med at lave spillet. Derfor har jeg lavet 3 krav som der skal være med i spillet, for at selve spillet vil fungere på en basic måde.

Krav 1

Det første krav til spillet, lyder på at røveren i spillet, skal kunne finde ud af at slippe væk fra det politi, som er efter røveren når personen er på vej væk med alle tyvekosterne. Det er derfor vigtigt at røveren slipper væk med tyvekosterne, så røveren slipper for fængselsstraf og får en profit.

Krav 2

Det andet krav til spillet, lyder på at politiet skal kunne forudsige hvilken rute røveren vil bruge som flugt vej, så de kan i møde komme røveren før røveren, slipper væk, så politiet kan sætte røveren i fængsel så røveren ikke får profit fra de stjalne varer.

Krav 3

Det tredje krav til spillet, lyder på at spillet skal have en GUI (Graphical User Interface), som viser politiet, røveren, hvilken runde som spillet er i gang med, hvor mange gange politiet eller røveren har vundet og hvor mange procent politiet eller røveren har vundet ud af alle runderne, så man har en form for statistik. Samtidigt med det, så skal politiet og røveren have forskellige farver, så man nemt kan se hvem der har vundet et hvis antal gange.

Kravanalyse

Krav 1

For at røveren vil kunne slippe væk fra politiet, så kræver det at røveren kender til gennemsnitspositionen (massemidtpunktet) af alle politibetjentene. Ud fra massemidtpunktet, så kan røveren finde ud af hvilken retning som ville give størst mulig chance for at slippe væk fra politiet.

```
150 def moveRobber():
151     global COM
152     #Udregn massemidtpunkt for politiet.
153     #COM = Center of Mass
154     COM = PVector()
155
156     totalWeight = 0
157     for cop in cops:
158         w = (1/dist(cop.pos.x, cop.pos.y, robber.pos.x, robber.pos.y))*2.2
159         totalWeight += w
160         COM.x += cop.pos.x * w
161         COM.y += cop.pos.y * w
162
```

Figur 1, moveRobber() funktionen

Massemidtpunktet bliver udregnet ved at finde hver politimands position, figur 1 linje 157 - 161

derefter vil man så beregne afstanden til massemidtpunktet fra røverens position. På figur 2 linje 168 sættes afstanden til at være røverens position, hvor afstanden så bliver trukket fra COM som er massemidtpunktets position af politiet. Derefter skal røveren så finde ud af hvor langt den skal rykke sig, og det gøres ved at gange afstanden mellem røveren og massemidtpunktet, med røverens bevægelses hastighed. Det sker i linje 173 efter at afstanden, som er en vektor bliver lavet til en enhedsvektor i linje 172 på figur 2. Til sidst skal røveren så bevæge sig til den nye position, det sker ved at tilføje afstanden til røverens position se linje 176 på figur 2.

```
166 #Find retningen væk fra COM.
167 #Beregn afstanden til massemidtpunktet
168 afstand = robber.pos.copy()
169 afstand.sub(COM)
170
171 #Vælg hvor langt røveren skal bevæge sig
172 afstand.normalize()
173 afstand.mult(robber.speed)
174
175 #Flyt røveren
176 robber.pos.add(afstand)
177 robber.last = afstand
```

Figur 2, moveRobber() funktionen

Røveren bevæger sig altså nu væk fra politiet, hvilket den skulle i krav 1.

Krav 2

For at kunne få politiet til at kunne forudsige røverens position så de har nemmere ved at fange ham, så kræver det at de faktisk kender den retning som det ville give mest mening for røveren at bevæge sig i.

```
137 def moveCops():
138     #Politiet bevæger sig imod det sted hvor røveren er.
139     #Det er nemlig sådan politiet gør.
140     for cop in cops:
141         afstand = robber.pos.copy()
142         afstand.sub(cop.pos)
143         afstand += robber.last * dist(cop.pos.x, cop.pos.y, robber.pos.x, robber.pos.y)
144         #Sørg for at politiet kun bevæger sig 1 pixel
145         afstand.normalize()
146         afstand.mult(cop.speed)
147
148         cop.pos.add(afstand)
```

Figur 3, moveCops() funktionen

For at politiet kan finde ud af hvor røveren er, så skal de finde røverens position ud fra afstanden af hver politimands position til røverens position. Derefter laver man afstanden om til en enhedsvektor og ganger så med politiets hastighed. Afstanden som nu også er ganget med politiets hastighed, sættes nu ind i politiets position og de vil derfor begynde at gå hen mod røveren. Se figur 3 linje 140-148.

Krav 3

For at kunne give mit spil et GUI, som viste de nødvendige ting, så skulle jeg finde ud af hvordan jeg ville have designet til at se ud. Eftersom jeg gerne vil have at man skal kunne se hvor mange gange tyv har vundet, hvor mange gange politiet har vundet, hvilken runde man er i gang med, hvor mange procent tyv og hvor mange procent politiet har vundet, så kræver det i alt 5 tekster som skal vises på skærmen.

```
86 def drawText():
87     global tyvVundet, copsVundet, tyvProcent, copsProcent, runde
88     fill(255,0,0)
89     textSize(15)
90     if runde > 0:
91         copsProcent = (float(copsVundet)/float(runde))*100
92         copsProcent = "{:10.2f}".format(copsProcent)
93         tyvProcent = (float(tyvVundet)/float(runde))*100
94         tyvProcent = "{:10.2f}".format(tyvProcent)
95         text("Tyv har vundet " + str(tyvVundet) + " gange", 5, 15)
96         text("Politiet har vundet " + str(copsVundet) + " gange", 5, 35)
```

Figur 4, drawText() funktionen



Billede 1, spilledets nuværende GUI

Måden de tekster bliver vist på skærmen er ved hjælp af min drawText() funktion som jeg har lavet. Denne funktion indeholder alt den kode som der er blevet brugt til at kunne få de 5 tekster op på skærmen. Teksten med "Tyv har vundet x gange", bliver lavet inde i funktionen med en Processing funktion som hedder text(). Funktionen text() tager imod en streng, x-koordinat og y-koordinat. Se figur 4 linje 95. Det er på samme måde alle de andre tekster bliver vist på skærmen.

For at ændre farven bruges fill() som er en funktion i Processing. Funktionen fill() modtager farve i rgb format og skal bruges lige før den text, box eller hvad man vil have

```
180 def drawCopsAndRobbers():
181     global COM, cops, robber, rudy, morten, minim
182
183     for cop in cops:
184         image(morten, cop.pos.x - 57, cop.pos.y - 57, 130,130)
185
186         image(rudy, robber.pos.x - 150, robber.pos.y - 50, 230,160)
```

Figur 5, drawCopsAndRobbers() funktionen

farvet, før objektet får den farve. Se figur 4 linje 88. Til sidst for at designe mit GUI skal teksten have en passende størrelse. Til det bruger jeg funktionen textSize(). Funktionen textSize() er en funktion i Processing som bruges til at sætte størrelsen på den text som man vil have vist. Her bruger jeg text størrelsen 15. Se figur 4 linje 89. Det sidste der mangler i min GUI, er politiet og røveren som skal være et billede. Politiet skal være et billede af Morten Spiegelhauer og røver skal være Rudy Frederiksen. Billederne er blevet sat ind ved at bruge Processings funktion, image(). Funktionen image() modtager navnet på filen som du vil bruge, x-koordinat, y-koordinat, bredde og højde. Se figur 5 linje 184.

Gennemgang af programmet

Opbygning af programmet

Flowdiagram

Inden flowdiagrammet bliver fortalt, så vil jeg lige snakke om, hvad man bruger et flowdiagram og hvorfor det er en god ide.

Det er en god ide at have et flowdiagram med i en synopsis eller rapport, da det er en nem måde at forklare hvad der sker i ens program, samt hvordan programmet er opbygget.

På figur 4 kan man se flowdiagrammet over mit program. Flowdiagrammet forklarer lige fra man starter programmet, til der er fundet en vinder i runden som der bliver spillet.

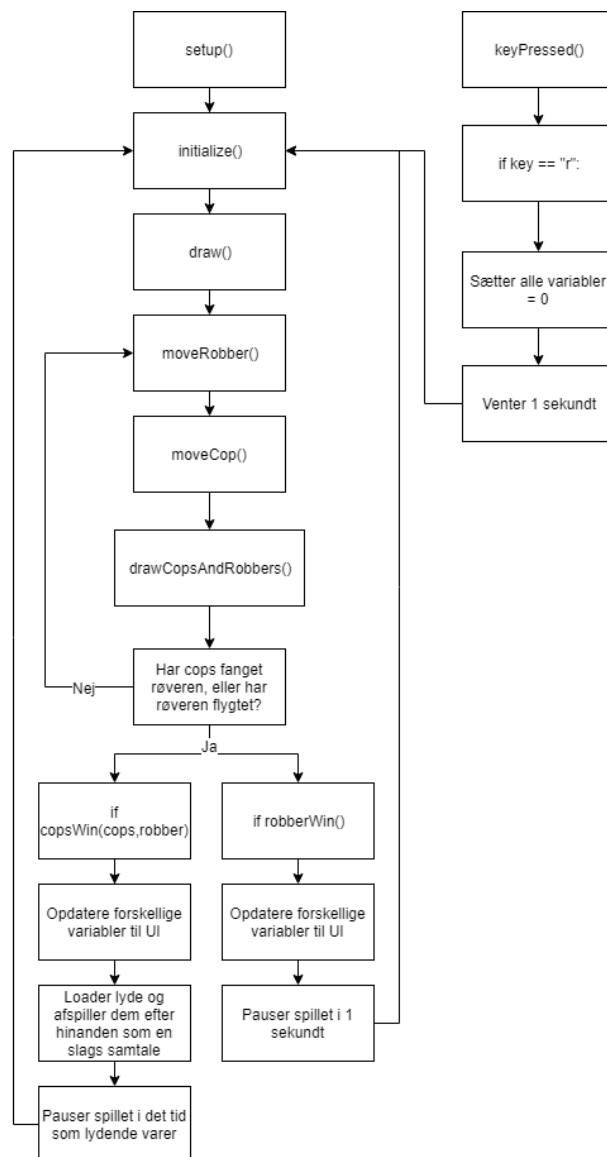
Sår man starter programmet, kører `setup()` først for at sætte alle de nødvendige variabler osv op, derefter skal `initialize()` køres så variablerne får de korrekte værdier. Efter `initialize()` er færdig, køres `draw()` som tegner alt hvad der skal vises på skærmen. `MoveRobber()` køres så for at rykke røveren og efter røveren har rykket sig, så køres `moveCop()` som rykker politiet. Derefter bliver der tjekket om røveren har flygtet eller om røveren er blevet fanget af politiet. Hvis ingen ting er sket, så rykker røveren og politiet igen, og det bliver de ved med indtil røveren flygter eller politiet fanger røveren.

Hvis det er politiet som fanger røveren, så bliver de forskellige variabler opdateret, så som runde bliver én større osv. Derefter bliver der afspillet en form for dialog mellem politiet (Morten) og røveren (Rudy) og imens det foregår, bliver programmet pauset. Når samtalen er færdig starter spillet forfra med en ny runde.

Hvis det derimod er røveren som flygter, så bliver de forskellige variabler opdateret som før, samtidigt med det så bliver spillet pauset i 1 sekund hvor spillet så starter en ny runde efter.

Der er tilsidst en funktion kaldet `keyPressed()` som tjekker om der bliver trykket på tastaturet. Hvis der bliver trykket på tastaturet og tasten er "r" så nulstilles alle variabler og spillet bliver pauset i 1 sekund hvor spillet så starter forfra.

Alt dette er vist grafisk på flowdiagrammet på figur 6.



Figur 6, flowdiagram over spillet

Klasser som der er brugt i spillet

Til spillet er der brugt to forskellige klasser, for at spillet kører som det skal. Der er klassen Cop og klassen Robber. Cop og Robber ligger i hver sin fil som hedder CopLib og RobberLib.

CopLib

CopLib indeholder som skrevet før, klassen Cop.

Cop er en klasse som definerer positionen, hastigheden, og farven på politiet som der er i spillet. Se linje 3 - 5 på figur 7.

Klassen indeholder en funktion som modtager et x-koordinat og et y-koordinat, ud fra det sætter funktionen x- og y-koordinaterne ind i en vektor som bliver gemt i variablen pos, som er politiets position. Politiets bevægelses hastighed sættes derefter til 1, så de bevæger sig i stedet for at stå stille, se figur 7 linje 8. Til sidst sættes farven på den lille prik som repræsenterer hver enkelt politimand. Her er farvekoden i RGB og farven er blå se figur 7 linje 9.

RobberLib

RobberLib indeholder sådan set det samme som CopLib. Forskellen mellem CopLib og RobberLib er at CopLib som sagt bliver brugt til politiet og RobberLib bliver så brugt til røverne. RobberLib indeholder dog en variabel mere end CopLib gør, hvilket er variablen "last". Last variablen starter med at være tom, se figur 8 linje 5, og når man så kalder RobberLib i spillet, så sættes "last" til en vektor med koordinaterne (0,0), se figur 8 linje 11. "last" variablen bliver så i spillet ændret fra 0 vektoren, til en vektor som indeholder den afstand der er mellem politiet og røveren, se figur 9 linje 177 da det skal bruges til at gøre så røveren kan undvige politiet og derved undgå at blive fanget.

```
1 class Cop:
2     pos = None
3     speed = 0
4     col = None
5
6     def __init__(self, x, y):
7         self.pos = PVector(x,y)
8         self.speed = 1
9         self.col = color(33,198,255)
```

Figur 7, CopLib klassen

```
1 class Robber:
2     pos = None
3     speed = 0
4     col = None
5     last = None
6
7     def __init__(self, x, y):
8         self.pos = PVector(x,y)
9         self.speed = 1
10        self.col = color(255,33,33)
11        self.last = PVector(0,0)
```

Figur 8, RobberLib klassen

```
168 afstand = robber.pos.copy()
169 afstand.sub(COM)
170
171 #Vælg hvor langt røveren skal bevæge sig
172 afstand.normalize()
173 afstand.mult(robber.speed)
174
175 #Flyt røveren
176 robber.pos.add(afstand)
177 robber.last = afstand
```

Figur 9, robber.last variablen i brug

Biblioteker som der er brugt

Random

Random er et bibliotek som der findes til Python. Biblioteket indeholder mange forskellige funktioner til at finde et tilfældigt tal eller objekt som der enten er i en liste, eller som er nummereret. I mit spil bliver random biblioteket brugt til at hente en funktion ind, som hedder randint. Randint bruges her til at finde et tilfældigt tal mellem x og y som man selv skal definere. I spillet bruges funktionen til at finde hvilken sætning Rudy (Røveren) skal sige når han bliver fanget.

Her finder den et tilfældigt tal i mellem 0 og antal elementer som der er i listen som indeholder Rudy's sætninger. Se figur 10 linje 74.

Time

Time er et bibliotek som der findes til Python ligesom random. Time er et bibliotek som der blandt andet bliver brugt til at kunne "pause" hele ens program i et vis antal sekunder, som man selv kan vælge. I mit spil bliver time her brugt til at give en funktion som hedder sleep. Sleep bliver i mit spil brugt til at pause spillet

imens der er en "dialog" imellem Morten(politi) og Rudy(røveren), så man ikke er i gang med at spille imens, da lyden så vil køre dobbelt, som er en dårlig oplevelse. Se figur 10 linje 79

Minim

Minim er et bibliotek som man kan tilføje i Processing. Biblioteket gør det muligt at bruge lyd i sit program man har lavet i Processing. Minim bliver i mit program brugt til at føre dialogen mellem politiet og røveren. Man starter med at loade filen som man vil afspille, derefter afspilles lyden og stopper igen når den er slut. Se figur 10 linje 70-71 for brug af Minim.

```
66 if copsWin(cops, robber):
67     #running = False
68     copsVundet = copsVundet + 1
69     runde = runde + 1
70     fs=minim.loadFile("HvadPokkerErDetDuHarGangI.mp3")
71     fs.play()
72     lende = float(fs.length())/1000
73     time.sleep(lende)
74     i = randint(0,len(Rudy_lyd)-1)
75     sf=minim.loadFile(str(Rudy_lyd[i]))
76     sf.play()
77     langde = float(sf.length())/1000
78     print("Cops wins")
79     time.sleep(langde)
80     initialize()
```

Figur 10, brug af randint

Test af programmet

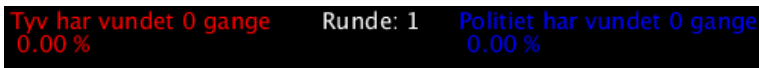
Hvordan fungerer det?

I spillets nuværende stadie, fungerer alle tingene som de skal. Politiet fanger røveren, har dialog med røveren og spillet starter forfra, hvor teksten på skærmen bliver opdateret fra "Politiet har vundet 0 gange" til "Politiet har vundet 1 gange", samtidigt bliver politiets og røverens procenter som de har vundet opdateret, og runden som man er i gang med bliver 1 større. Som det er lige nu, er der 10 politimænd som skal fange røveren. Røveren undviger dog nogle gange politiet ved at lave en lille finte med dem. Dog er det mest politiet der vinder, ved at de når og gå ind foran røveren da de kan forudsige vejen som røveren vil tage. Som programmet er lige nu, er det fuldstændigt spilbart og har ikke rigtig nogle fejl som der lægges mærke til.

Overholdes kravene?

I spillets nuværende stadie, overholdes alle de 3 krav som jeg her stillet til spillet. Røveren kan undvige politiet, politiet kan forudsige røverens flugt vej og GUI'en fungerer og

opdatere som den skal. Teksten med "Politiet har vundet x gange", kunne måske gøres i en lidt lysere blå, da den nuværende godt kan blende sig ind med den sorte baggrund. Se billede 2.



Billede 2, billede af den blå tekst som der blandes sammen med den sorte baggrund

Beskrivelse af arbejdsprocessen

Jeg startede med at få udleveret et program, hvor politiet og røveren var streger af blå og rød, det var meget tilfældigt hvem der vandt de forskellige runder. Jeg har så sidenhen tilføjet lyd, billeder, tekst på skærmen, gjort røveren bedre, gjort politiet bedre og en mulighed for restart af spillet.

Det har været lidt svært at finde ud af matematikken bag måden om politiet og røveren bevægede sig da det er en del som der allerede er lavet. Derfor har meget af tiden gået med at researche programmet for at finde ud af hvordan det er sat sammen og hvordan det fungere.

Bilag

Bilag 1 Cops and robbers

```
add_library('minim')
from CopLib import *
from RobberLib import *
import time
from random import randint

def setup():
    size(500,500)
    frameRate(100)
    global cops, tyvVundet, copsVundet, tyvProcent, copsProcent, runde, rudy, morten, minim,
    Rudy_lyd, Morten_lyd
    minim=Minim(this)
    tyvVundet = 0
    copsVundet = 0
    tyvProcent = 0
    tyvProcent = "{:10.2f}".format(tyvProcent)
    copsProcent = 0
    copsProcent = "{:10.2f}".format(copsProcent)
    runde = 0

    rudy = loadImage("Rudy.png")
    morten = loadImage("Morten.png")
    Rudy_lyd = ["SmartKomplotDuDygtig.mp3", "Retssag.mp3", "Nej.mp3",
    "JamenDetErJolkkeSandt.mp3", "DuBeskylderMigForEtEllerAndet.mp3",
    "DetLyderVeldigSmartDenHistorie.mp3", "DetLyderSomEnFantastiskHistorieDetDer.mp3",
    "DetErJolkkeSandt.mp3", "DetErFaktiskEnMusikvideo.mp3",
    "BeskyldningerSomJegIkkeVedHvadJegSkalGoreVed.mp3"]
    Morten_lyd = "HvadPokkerErDetDuHarGangl.mp3"

    cops = list()

    initialize()

def initialize():
    #I denne funktion skal hele spillet
    #startes forfra.
```

global cops, robber

```
#running = True
#Baggrunden cleares
background(0)
```

```
#Ny position til røveren
robber = Robber(width/2, height/2)
```

```
#Nye positioner til politiet.
#Tøm listen
cops[:] = []
#og tilføj nye politimænd
for i in range(0, 10):
    x = random(10, width - 10)
    y = random(10, height - 10)
    cops.append(Cop(x,y))
```

```
def draw():
    global cops, robber, running, tyvVundet, copsVundet, runde
    #if running == True:
    moveRobber()
    moveCops()

    if robberWin():
        #running = False
        tyvVundet = tyvVundet + 1
        runde = runde + 1
        x = random(10, width - 10)
        y = random(10, height - 10)
        print("Robber wins")
        time.sleep(1)
        initialize()

    if copsWin(cops, robber):
        #running = False
        copsVundet = copsVundet + 1
        runde = runde + 1
        fs=minim.loadFile("HvadPokkerErDetDuHarGangl.mp3")
        fs.play()
        lende = float(fs.length())/1000)
        time.sleep(lende)
```

```
i = randint(0,len(Rudy_lyd)-1)
sf=minim.loadFile(str(Rudy_lyd[i]))
sf.play()
langde = float(sf.length())/1000
print("Cops wins")
time.sleep(langde)
initialize()
```

```
background(0)
drawCopsAndRobbers()
drawText()
```

```
def drawText():
    global tyvVundet, copsVundet, tyvProcent, copsProcent, runde
    fill(255,0,0)
    textSize(15)
    if runde > 0:
        copsProcent = (float(copsVundet)/float(runde))*100
        copsProcent = "{:10.2f}".format(copsProcent)
        tyvProcent = (float(tyvVundet)/float(runde))*100
        tyvProcent = "{:10.2f}".format(tyvProcent)
        text("Tyv har vundet " + str(tyvVundet) + " gange", 5, 15)
        text("    " + str(tyvProcent) + " %", -20, 30)
        fill(0,0,255)
        textSize(15)
        text("Politiet har vundet " + str(copsVundet) + " gange", 300, 15)
        text("    " + str(copsProcent) + " %", 277, 30)
        fill(255)
        text("Runde: " + str(runde + 1), 210, 15)
```

```
def keyPressed():
    global tyvVundet, copsVundet, tyvProcent, copsProcent, runde
    if key == "r":
        tyvVundet = 0
        copsVundet = 0
        tyvProcent = 0
        tyvProcent = "{:10.2f}".format(tyvProcent)
        copsProcent = 0
        copsProcent = "{:10.2f}".format(copsProcent)
        runde = 0
        time.sleep(1)
        initialize()
```

```
def robberWin():
    if robber.pos.x < 0:
        return True
    if robber.pos.x > 500:
        return True
    if robber.pos.y < 0:
        return True
    if robber.pos.y > 500:
        return True
    return False

def copsWin(cops,robber):
    for cop in cops:
        if dist(cop.pos.x, cop.pos.y, robber.pos.x, robber.pos.y)<1:
            #running = False
            return True
    #Denne funktion skal returnere True,
    #hvis politiet har fanget røveren
    return False

def moveCops():
    #Politiet bevæger sig imod det sted hvor røveren er.
    #Det er nemlig sådan politet gør.
    for cop in cops:
        afstand = robber.pos.copy()
        afstand.sub(cop.pos)
        afstand += robber.last * dist(cop.pos.x, cop.pos.y, robber.pos.x, robber.pos.y)
        #Sørg for at politiet kun bevæger sig 1 pixel
        afstand.normalize()
        afstand.mult(cop.speed)

        cop.pos.add(afstand)

def moveRobber():
    global COM
    #Udregn massemidtunkt for politiet.
    #COM = Center of Mass
    COM = PVector()

    totalWeight = 0
    for cop in cops:
```

```
w = (1/dist(cop.pos.x, cop.pos.y, robber.pos.x, robber.pos.y))**2.2
totalWeight += w
COM.x += cop.pos.x * w
COM.y += cop.pos.y * w
```

```
COM.x /= totalWeight
COM.y /= totalWeight
```

```
#Find retningen væk fra COM.
#Beregn afstanden til massemidtpunktet
afstand = robber.pos.copy()
afstand.sub(COM)
```

```
#Vælg hvor langt røveren skal bevæge sig
afstand.normalize()
afstand.mult(robber.speed)
```

```
#Flyt røveren
robber.pos.add(afstand)
robber.last = afstand
```

```
def drawCopsAndRobbers():
    global COM, cops, robber, rudy, morten, minim

    for cop in cops:
        image(morten, cop.pos.x - 57, cop.pos.y - 57, 130,130)

    image(rudy, robber.pos.x - 150, robber.pos.y - 50, 230,160)
    fill(255)
    ellipse(COM.x, COM.y, 1, 1)

    fill(robber.col)
    ellipse(robber.pos.x, robber.pos.y, 1, 1)

    for cop in cops:
        noStroke()
        fill(cop.col)
        ellipse(cop.pos.x, cop.pos.y, 1, 1)
```

Bilag 2 CopLib

```
class Cop:
```

```
pos = None  
speed = 0  
col = None
```

```
def __init__(self, x, y):  
    self.pos = PVector(x,y)  
    self.speed = 1  
    self.col = color(33,198,255)
```

Bilag 3 RobberLib

```
class Robber:
```

```
    pos = None  
    speed = 0  
    col = None  
    last = None
```

```
def __init__(self, x, y):  
    self.pos = PVector(x,y)  
    self.speed = 1  
    self.col = color(255,33,33)  
    self.last = PVector(0,0)
```