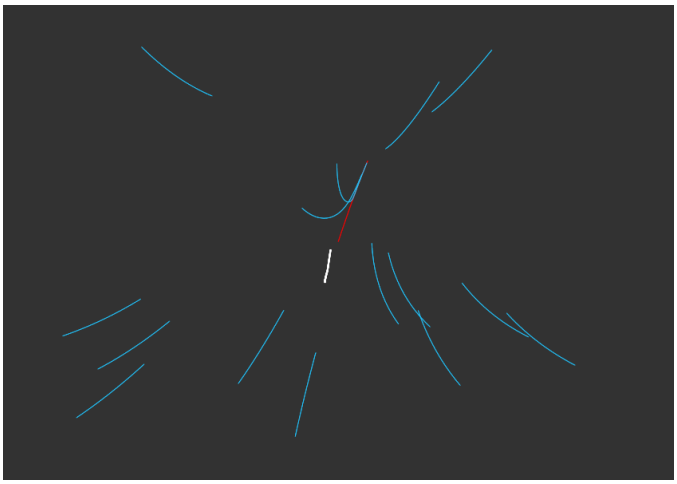
Årsprøveprojekt i programmering B
2.d

Forår 2019

Til sommer skal vi holde en årsprøve i programmering, og for at forberede jer til eksamen, forsøger vi at få årsprøven til at ligne den rigtige eksamen så meget som muligt. Derfor laver vi et projekt og en rapport, som beskriver hvad I har lavet. Det danner grundlag for evalueringen til årsprøven.

I dette projekt skal vi arbejde med et lille spil, der hedder "Cops and Robbers". På dansk svarer navnet vel til røvere og soldater, som de fleste har leget som barn. Spillet blev udviklet i starten af 1980'erne[1], og stammer egentlig fra en gren af matematikken der hedder *grafteori*[2].

Vi skal bruge spillet til at prøve at anvende nogle af de ting vi har arbejdet med i timerne indtil nu, og til sidst skal vi også skrive en lille rapport, som vi kan bruge til årsprøverne.



Figur 1: 15 blå politimænd jagter en rød røver.

[1] *R. Nowakowski and P. Winkler, Vertex to vertex pursuit in a graph, Discrete Math. 43 (1983), 235239.* og *A. Quilliot, A short note about pursuit games played on a graph with a given genus, J. Combin. Theory Ser. B 38 (1985), 8992.*

[2] Ikke den slags grafer som I kender fra funktionsbegrebet! Læs mere på http://en.wikipedia.org/wiki/Graph_theory

Spillet

Spillet består af en røver, der har en position (x, y) på skærmen, og et antal politimænd –ligeledes med hver deres position på skærmen– der skal forsøge at fange røveren. Spillet kører efter tur, så røveren bevæger sig først, en pixel i valgfri retning, og derefter må alle politimændene bevæge sig. I vores tilfælde kan spillet stoppe på to måder: Enten ved at røveren undslipper, ved at nå kanten af vinduet, eller ved at en politimand stiller sig på samme pixel som røveren, og fanger ham.

Øvelse 1. Den første opgave handler om at lære koden at kende. Kig på koden, og prøv at forstå hvilke elementer programmet består af.

- For at få spillet til at starte forfra, kan man kalde funktionen `initialize()`. Lav en `keyPressed()`-funktion, og genstart spillet hvis der bliver klikket på `r`.
- Spillet består i af to slags aktører: Politimænd og røveren. De to aktører er beskrevet i hver sin klasse, `Cop` og `Robber`. Hvad en klasse er, og hvordan man kan bruge sådan en i et program, er en længere historie, og vi vender tilbage til teorien senere. Lige nu kan du bare tænke på en klasse, som en model af noget der findes i virkeligheden. En *datamodel*. Prøv at kigge i koden, og find alle de steder hvor klasserne bliver brugt. Hvilke variable bliver der oprettet, og hvordan bruges de i programmet? Diskutér med sidemanden hvilke fordele der kan være, ved at bruge disse klasser i programmet.
- I `draw`-funktionen er programmets væsentligste struktur opbygget. Undersøg koden, og tegn et flowdiagram, der viser hvordan programmet afvikles.
- De to funktioner `robberWin` og `copWin` skal bruges til at afgøre, om spillet er afsluttet. Prøv om du kan få funktionerne til at returnere `True`, hvis røveren slipper væk fra skærmen, eller hvis en politimand fanger røveren.
- Fra `draw`-funktionen bruges `robberWin` og `copWin` i et par `if`-sætninger. Her skal programmet genstartes, hvis betingelsen er opfyldt. Udfyld evt. med anden funktionalitet, der ville være oplagt at udføre på dette tidspunkt.

Røverens strategi

Der er mange politifolk efter røveren, så han skal bruge en ret god strategi, for at kunne slippe væk. Heldigvis har han haft fysik A i gymnasiet, så han har regnet ud, at i stedet for at stikke af fra hver enkelt politimand, så kan han bare stikke af fra politifolkenes gennemsnitlige position! Som du kan se, når du kører programmet, så virker strategien nogenlunde, men der er bestemt plads til forbedring.

En effektiv metode til at forbedre røverens strategi er at vægte politimændene forskelligt, når deres gennemsnit udregnes. Det kalder man et *vægtet gennemsnit*. Normalt gennemsnit udregnes ved formlen

$$\bar{x} = \frac{\sum_{i=0}^{n} x_i}{n}$$

hvor n er antallet af politimænd, og x_i er positionen for en politimand. (Tilsvarende for y) Det vægtede gennemsnit findes ved

$$\bar{x} = \frac{\sum_{i=0}^{n} w_i \cdot x_i}{\sum_{i=0}^{n} w_i}$$

hvor w_i er et tal, der er højt hvis politimanden er tæt på røveren, og lille hvis politimanden er langt væk.

Øvelse 2. Prøv at ændre i funktionen `moveRobber`, så den bruger et vægtet gennemsnit i stedet. Hvilken udregning kan du lave i programmet, så du får en passende værdi af tallet w_i?

Hint: Tænk på det som en funktion w(d), hvor d er afstanden, og w er vægten. Hvordan vil du gerne have, at denne funktion opfører sig?

Øvelse 3. Kan du finde andre måder at forbedre røverens strategi?

Politiets strategi

Røveren har haft fysik A i gymnasiet, men adgangskravene til uddannelsen som politibetjent stiller ingen krav om naturvidenskabelige fag. Politimændene har derfor valgt en ret dårlig strategi til at fange røveren. De bevæger sig allesammen i retning af røverens aktuelle position, i stedet for at samarbejde, eller forsøge at forudsige hvor de kan komme røveren i forkøbet. Som du kan se, lykkes det ind imellem for dem at fange røveren alligevel, men de har i den grad brug for hjælp.

Øvelse 4. Udtænk en bedre strategi for politiet, i funktionen `moveCops`. De kan enten samarbejde, eller hver især vælge en bedre rute, når de skal fange røveren. Beskriv din løsning i tekst, evt. med en algoritme, der skridt for skridt forklarer hvad politiet skal gøre. Forsøg herefter om du kan skrive kode, der får politiet til at følge din strategi.

Fri opgave

Når du er færdig med røveren og politiets strategier, kan du lave forskellige andre forbedringer til programmet.

- Lav programmet dynamisk, så man for eksempel kan klikke med musen på skærmen, og så flytter røveren sig derhen hvor man har klikket. Eller måske en af politimændene?
- Lav noget federe grafik.

Rapportering

Du skal skrive en lille rapport om dit arbejde. Du skal skrive et kort afsnit til henholdsvis røverens og politiets strategi, hvor du forklarer om de tanker du har gjort dig, og viser din løsning. Du skal bruge relevante screenshots fra programmet, og du skal forklare om den kode du har skrevet.

Rapporten skal ikke være længer end 10 sider, ligesom til eksamen. Det er vigtigt at du tænker over hvilke ting du vil fortælle om i rapporten og til årsprøven, da de er med til at give et billede af dit faglige niveau.

Tænk over hvilke abstrakte dokumentationsformer du kan bruge, til at fortælle om dit program. Jo mere du kan vise med grafer, figurer, flowdiagrammer, osv., jo nemmere bliver det at forstå hvad du har lavet i projektet.

Til sidst skal **hele** koden fra programmet vedhæftes rapporten.

Her er et forslag til en opbygning af rapporten

1. Indledning
2. Røverens strategi
   (a) Forklaring af din idé
   (b) Præsentation af løsningen
3. Politiets strategi
   (a) Forklaring af din idé

    (b) Præsentation af løsningen

4. Evt. udvidelser af programmet
5. Test af programmet
6. Konklusion
7. Bilag
    (a) Hele koden.

Vi bruger ca. 7 moduler i klassen på projektet, og resten forventes det af du laver på egen hånd. Den endelige afleveringsdato aftaler vi i klassen. Hold øje med portfolio!

Til årsprøven får du 24 minutter til at:

- Præsentere dit arbejde i projektet
- Fortælle om en ukendt opgave, som du har arbejdet med i 24 minutters forberedelse inden prøven.