Luleå tekniska universitetet

Skellefteå

Laboration 1: FSM i Python 3

S0006D, Datorspels AI

Laboration: 1

Oscar Östryd

Oscstr-9@student.ltu.se

https://github.com/oscstr-9/Datorspels-AI---S0006D.git

31/01-2021

Innehållsförteckning

Problemspecifikation

Uppgiften för denna laboration var att skapa en finite state machine (FSM) som skulle kunna arbeta, handla, äta, dricka, sova, och umgås med kompisar. Kraven var att det skulle finnas två olika arbetsplatser, agenterna inte fick ”dö” och meddelanden skulle kunna skickas mellan agenterna för att boka in en tid att mötas upp, detta bokade möte skulle även kunna avbokas ifall en agent var i stort behov av mat eller dryck. Det skulle finnas någon sort ”realistisk” timer som bestämde när agenterna behövde olika saker. Denna uppgift skulle då skapas i Python 3. Denna uppgift hade lite olika krav på vilket betyg som man siktade på. Jag anser att mitt projekt uppnått betyg 3.

Användarhandledning

För att kunna test köra laborationen behöver du packa upp den zippade projekt filen (oscstr9_Lab1.zip). Efter det bör det finnas en fil som heter Main.py, öppna denna fil och testa projektet. En kommandotolksruta kommer att öppnas och be dig ange hur många agenter som ska finnas samt hur snabbt tiden ska gå. Det rekommenderas att det finns minst 4 agenter och en tidsmultiplikator på minst 720.

Algoritmbeskrivning

Projektet består av tre filer, en person fil, en state fil och en main fil.

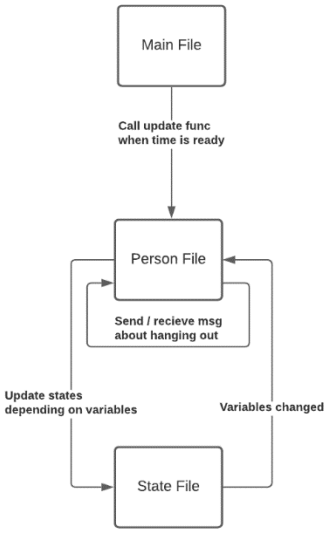
Main filen är den som startar programmet och håller koll på tiden och meddelar agenterna om när de behöver uppdateras. Den är väldigt enkel med vad den gör och är nästan som sin egen del.

Person filen har alla värden för varje agent samt en funktion för att skicka meddelanden och en för att ta emot meddelanden. Den har dessa funktioner för att det ska vara enkelt att veta just vem jag skickar meddelanden till och vad varje agent ska göra med svaret de får tillbaka.

State filen består av alla states som en agent kan vara i. Varje state är en class med en exekveringsmetod och en metod för att få reda på vilket state en agent är i. detta skrevs på ett sådant sätt för att man då enkelt kan ge ett state till en agent så att agenten själv kan hålla koll på allt om sig själv och ingen annan.

Systembeskrivning

Person klassen fungerar som ett objekt som håller all information för en agent och ändrar inga värden själv förutom när den får ett meddelande som säger åt den att göra det. Detta gör det enkelt att veta hur värdena för en agent borde ändras beroende på dess state. State filen består av många klasser där varje klass är ett state som kan ges till en agent för att bestämma ändringar på värden och vilket state som agenten borde byta till när den inte längre är i behov av det state den befinner sig i just nu. Att tydligt separera vad varje fil och varje klass får göra och ändra på är en viktig del som gör projektet enklare att läsa och förstå.



Lösningens begränsningar

Min lösning har risken att arbetsvärdet kan bli noll vilket jag inte anser vara en så farlig sak. Uppgiften nämner att en agent måste äta och dricka när den behöver så att den inte svälter eller törstar ihjäl, att arbetsvärdet når noll ser inte jag som något som har ihjäl agenten. Om tanken är att inga värden får nå noll skulle det stå tydligare i beskrivningen av uppgiften.

Diskussion

I början av uppgiften hade jag lite svårt att komma på en bra lösning för att uppdatera alla agenter samtidigt som jag ville ge dem friheten att kunna ha någon sorts kommunikation med varandra och valde till sist den som används nu. Denna lösning är inte den bästa för att skicka meddelanden då jag skulle kunnat göra en egen class som hanterar meddelanden. Lösningen undviker att låta någon agents värden att nå noll men har inget stop ifall det skulle hända vilket är något som i efterhand är svårt att få in på ett bra sätt.

Laborationen i allmänhet tyckte jag var ganska intressant och rolig att arbeta på då det är mycket att fundera. Laborationen hade även väldigt bra riktlinjer samtidigt som den gav mycket frihet till programmeraren.

Magnus Lindh, en klasskamrat, har varit till stor hjälp när det kommer till att svara på generella frågor samt att diskutera med om hur en lösning skulle kunna göras.

Online har jag fått en del hjälp med Python 3.9 syntax då jag inte arbetat med det på länge och börjat glömma bort och blanda ihop en del med andra språk.

Testkörningar

I mina tester har jag funnit många problem som jag löst och som det står just nu hittar jag bara ett möjligt problem vilket är att arbetsvärdet kan gå ner under noll. Att detta händer är väldigt otroligt, men eftersom alla värden som förändras är slumpmässigt bestämda mellan 40 – 80 när en agent skapas finns det en möjlighet att om alla värden hamnar under ca. 42 kan agentens arbetsvärde sjunka under noll innan agenten får möjlighet att öka detta värde.

Annars under testerna som gjorts har jag tittat noga på att meddelandena skickas och tas emot korrekt vilket var ett problem i början samt att alla värden ändras som de ska när de ska ändras.