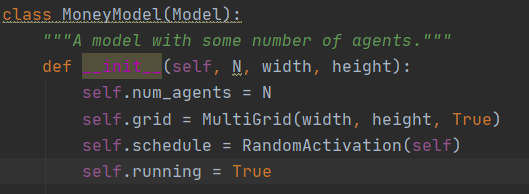
**1: King of the mesa**

King of the Mesa is een agent based programma, je kan het gebruiken om om snel agent-based modellen te maken. Hij heeft ingebouwde functies voor bijvoorbeeld grids en agent schedulers. Wat ik merkte tijdens het doorlopen van de tutorial is dat je de data die je uit de simulatie krijgt makkelijk kan omzetten door gebruik te maken van data analyse tools die gemaakt zijn voor python, bijvoorbeeld panda’s en dat vind ik een duidelijk voordeel. Ik kreeg het niet voor elkaar om de agents live te simuleren zoals je ziet in bijvoorbeeld unity dus dat kan een nadeel zijn maar misschien heb ik niet goed genoeg geprobeerd. Zelf heb ik een functie toegevoegd genaamd, give\_more\_money deze lijkt op de give money functie maar hier terecht te komen moet je een wealth boven de 5 hebben en geef je 2 weg.

**2: Omschrijf abstract een stateful agent**

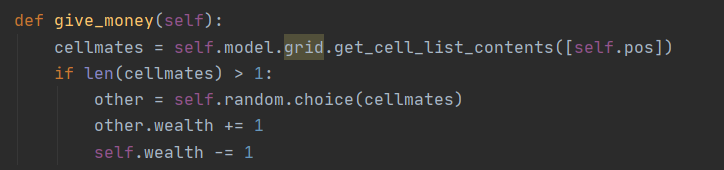
***2.1 initiële staat***

Dit is de initiële staat van het model zie je hier onder, hier geef je de hoeveelheid agents en de grote van het model mee. Deze waardes kun je bepalen voordat je hem runt en daarom lijkt dit mij de initiële staat en niet waar je de waardes invult want daar twijfelde ik eerst ook over.



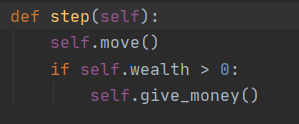
***2.2 see***

Hier staat de functie give\_money beschreven. In deze functie checkt hij de staat van het vakje waar hij zich bevindt, hij checkt namelijk hoeveel er in het vakje staan en kiest dan een agent om z’n geld aan te geven.



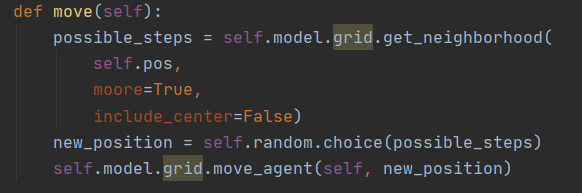
***2.3 act***

Tijdens deze stap beweegt de agent en acts. Hij voert namelijk de functie give\_money uit als hij genoeg geld heeft, tijdens die functie checkt hij pas of er genoeg andere agents in zijn vakje staan om zijn geld aan te geven.



***2.4 update***

Dit kun je zien als de update functie, elke keer dat hij naar een nieuw vakje gaat voert hij de acties opnieuw uit, hij beweegt checkt of zijn wealth hoger is dan 0 en als dat zo is geeft hij 1 weg en beweegt hij weer naar een ander vakje.



**3 Beschrijf je omgeving op basis van de dichotomies**

In de tutorial die ik gevolgd heb is de accessibility het vakje waarin hij staat. Hij kan namelijk alleen in zijn eigen vakje om zich heen checken om te checken of er andere agents op het zelfde vakje staan. Als je het versimpeld is dit een deterministische omgeving want als hij een stap zet naar een volgend hokje geeft hij wel of geen geld weg en ontvangt hij wel of geen geld. Het is alleen wel random welke agent het geld krijgt waardoor de volgende staat van de agents in het hokje gezien kan worden als non-deterministisch. Om te bepalen of het episodic of non-episodic heb ik gekeken naar de acties van de agent en die hebben wel degelijk invloed op zijn volgende stap want hij kan geen geld weg geven als hij dat niet heeft en hij kan geen geld ontvangen als hij in een vakje staat met een agent die geen geld heeft, hij is dus afhankelijk van de acties van andere agents en van wat hij zelf heeft gedaan in eerdere stappen maar daar denkt niet na over het effect van zijn volgende stap. Daarom zou ik zeggen dat het episodic is. De omgeving is static want er wordt een rooster met hokjes aangemaakt waar ze over bewegen en verder niks. Er is een kans dat er een moment is dat een agent uiteindelijk al het geld bezit maar dan stopt het model nog niet want in de stap erna geeft hij weer eentje weg en dan gaat het weer verder en daarom zou je het oneindig kunnen laten lopen wat het continuous maakt.

**4 Bedenk een voorbeeld waarbij minimaal 3 dichotomies**

Om bij het voorbeeld van een verkeerssituatie te blijven kwam ik met de situatie bij een kruising. Stel je visie wordt geblokkeerd door een vrachtwagen die naast je staat maakt dat de enviorment inaccessable omdat je dingen niet meer ziet die je anders wel zou zien. Ook zorgt dit ervoor dat de omgeving dynamisch is omdat die wordt beïnvloed door andere agents. Hier zou je niet mee te maken hebben wanneer de omgeving static zou zijn. De keuze je in de situatie zonder zicht maakt zou dan gezien kunnen worden als non-deterministisch want je kan oversteken en dan wel of niet geraakt worden. Maar in een situatie zonder andere agents kan je oversteken wanneer je wilt want je wordt toch niet geraakt dus er maar een gevolg wanneer je oversteekt en dat is dat je heelhuids aan de overkant komt. Dit soort keuzes hebben natuurlijk invloed op je resultaten, stel je wilt weten wat de doorstroming tijdens spitsuur is dan wil je het rijgedrag zo nauwkeurig mogelijk simuleren om te zien waar files ontstaan en wat de reden daarvan kan zijn. Dan kun je na de simulatie oplossingen bedenken om ophoping te voorkomen. Als je de slijtage van verf op het wegdek, bijvoorbeeld haaientanden, wilt simuleren dan kan dat heel simpeler door er bijvoorbeeld 100 auto’s overheen te laten rijden en kijken wat het effect daarvan is. Afhankelijk van de resultaten die je wilt kun je dus zelf bepalen hoe uitgebreid je de simulatie maakt. Het is daarom handig als je daar van te voren over nadenkt.