Ik heb ervoor gekozen om zelf een simulatie te maken met hulp van alle twee de netlogo tutorials. De file heet: burn_out_sim.logo, ik probeer het geluk van een groep studenten te simuleren op basis van hoe onzinnig de opdrachten zijn die zij krijgen, en hoeveel zij er per week krijgen. Te veel onzinnige opdrachten per week lijden uiteindelijk tot burnout van de student.

1. Volg de tutorial en omschrijf daarna in één paragraaf wat deze tool anders maakt dan andere programmeertalen, wat zijn de voor- en nadelen?

   Netlogo is anders dan Mesa in dat het een makkelijk te gebruiken UI heeft, i.p.v een console window. De UI is ook een stuk simpeler dan die van Unity. Daarnaast gebruikt Unity C# als programmeertaal en Mesa gebruikt python. Netlogo gebruikt een eigen programmeertaal, dit betekent dus dat het aan het begin even meer werk kan zijn om te leren werken met dit programma MAAR de programmeertaal die Netlogo gebruikt is relatief simpel om te leren.

   Voordelen:

   - Makkelijk om mee te beginnen
   - Duidelijke UI
   - Geïntegreerde documentatie
   - Makkelijk toevoegen van interactieve elementen zoals: knoppen, input velden, graphs

   Nadelen:

   - Erg verouderd vergeleken met de nieuwere versies van Unity
   - Beperkt in wat er mogelijk is qua visualisatie
   - Ziet er uit als iets uit de vorige eeuw (dat is het ook)

2.
   1. Mijn agents (studenten) hebben in mijn simulatie alleen maar een ‘burned-out’ of ‘not-burned-out’ state (in mijn code als ‘dead’ of ‘alive’)
   2. Ik heb werkelijk geen idee wat hier mee bedoeld wordt, ik kan wel het aantal agents in een specifieke staat opvragen en doe dat ook in mijn simulatie (om te kijken hoeveel studenten burned out zijn).
   3. Als een agent in een burnout state terechtkomt wordt dat gevisualiseerd als een kleurverandering van de agent naar rood
   4. Als mijn agent een bepaalde lage ‘happiness score’ bereikt dan wordt die in een burnout state gezet.

3. Beschrijf je omgeving op basis van de dichotomiën die hier op pagina 6 beschreven staan, en licht toe (dus niet alleen termen opsommen):

   1. Inaccessible, mijn environment is de echte wereld, dit is te breed om complete, accurate, informatie over te krijgen als agent.
   2. Non-deterministic, ik gebruik random nummer generators voor het simuleren van bijvoorbeeld start ‘geluk’ niveaus van studenten. Dit lijdt tot meerdere mogelijke effecten.

3. Episodic, besluiten worden alleen gebaseerd op situaties binnen de huidige episode
4. Dynamic, er zijn meerdere factoren, van buitenaf, die impact hebben op de ‘geluk’ niveaus van de student, zoals het aantal toetsen per week.
5. Continuous, mijn model is continuous omdat het een situatie probeert te simuleren die in principe onbegrensd is in mogelijke acties.

4. Het veranderen van mijn environment naar een statisch environment zorgt er alleen maar voor dat de simulatie toepasbaar is voor 1 specifiek scenario i.p.v. aanpasbaar voor meerdere scenario’s.

   Als mijn environment accessible zou zijn zou dat natuurlijk een grote positieve impact hebben om de nauwkeurigheid van mijn model, dingen zoals het weer, nieuws en duizenden andere factoren kunnen impact hebben op hoe gelukkig een student zicht voelt.

   Als ik mijn environment deterministisch zou maken moet ik i.p.v. keuzes door agents gebaseerd op random number generators, keuzes gebaseerd op de eigenschappen van de agents maken. Dit is meer realistisch maar ook gigantisch meer complex. Alles is deterministisch met perfecte informatie, maar wij hebben in dit scenario geen perfecte informatie. Je valt al snel in een discussie over ontologie/epistemologie met determinisme maar dat lijkt me een beetje boven het verwachte niveau van deze opdracht.