# Assessment 2: Documentation – GEOG5990M – Planning for Drunks

## Summary

This documentation provides an overview of the code used in the agent-based model (ABM) planning\_for\_drunks\_model.py. The following section provides an outline of where the code is sourced, the purpose of the model, and highlights errors encountered.

This project is part of the GEOG5990M module taught at the University of Leeds, the idea was provided by the course leader and is not my own.

## Purpose of the Model

The ABM simulates ‘drunk’ agents that are leaving a pub and attempting to return to their homes. There are 25 drunks who are assigned a number between 0-24, and are also assigned house numbers ranging from 10 to 250, going up in tens. Each drunken agent will leave the pub and move in random directions until they find their assigned home. As followed in the assessment brief, a drunk agent can only leave the pub if the other agent outside has arrived at their house. Agents cannot leave their homes once arrived, and the model will stop running until all 25 agents have returned home. The model has been coded to record how many times all of the agents have stepped on a coordinate; this has been done to provide data to create a density map. Agents can step on each other’s tracks, and each step increases the coordinate’s value by one.

## Code Source

There are three files for the model:

1. planning\_for\_drunks\_model.py
2. agentframework\_drunks.py
3. town.plan.txt

The first file contains the code for the model to run. This has been hand-coded by myself following the guides provided by Dr. Andy Evans on the GEOG5990M website: <http://www.geog.leeds.ac.uk/courses/computing/study/core-python/>

The second file contains my agent’s starting locations and variables, this has been hand-coded.

The third file includes the model’s environment data; it is a 300x300px raster file provided for the assignment, containing locations for agent homes, and a pub location.

Coding skills have also been developed during the module, through the completion of the futurelearn course “Learn to Code for Data Analysis”, online videos, and by reading guide to coding in Python.

## Model Guide

The following provides a guide to the code structure.

The first part of the models running script reads in the raster data to build the ABMs environment, this was provided in the town.plan.txt file. The second part reads into the agentframework\_drunks.py file to assign each agent a starting position and a function to navigate around the environment in random directions. The third part within the planning\_for\_drunks\_model.py file tells the agents to navigate around the environment and to stop once they have reached their home. A stepped\_environment.csv file was created based on the original ABM’s environment. As an agent steps on a coordinate, this is recorded down into the .csv and adds a value of 1 to show how many times a coordinate has been stepped on. Once the model has finished running, the .csv file is saved, and the model produces two figures; a figure of all agents at their assigned home (Figure 1), and a figure of a density map created using the model’s output data (Figure 2).

The code’s format is closely linked to- and based on- what I had written for assessment 1: <https://github.com/kmbutterfield/Programming-for-Spatial-Analysts-Assessment-1>

Assessment 1’s model was used as a foundation for creating functions, classes, and attributes. The assessment 2 code includes other types of code structures too, such as Booleans converted into integers.

##### 

Figure : Example image of agents reaching their homes from a model run

##### 

Figure : Example of density map creating during the model run based on figure 1

## Errors Encountered with the Model Code

A few errors were encountered and debugged during model creation. The first issue was that I wanted to record how many steps it took each agent to reach their home. I initially wrote my model in pseudo code but in the style of C#. In a mixture of coding languages, I managed to print how many steps it took each agent (Figure 3).

##### 

Figure : Code written from pseudo code to record number of steps

However, as I am still learning to code in Python, I struggled to find a Pythonic way to count the number of steps each agent took to get home.

Another issue I encountered was with recording the stepped data for the stepped\_environment\_output.csv file. Every time I ran the model, the .csv would fill with zeros. The issue turned out to be incredibly simple; I was typing “…\_x]=+1” instead of “…\_x]+=1 (Figure 4) as this made more logical sense to me.



Figure : Correct code to record steps

Ideally, I wanted the agents to know where they were going, and have a sense of direction. For example, if agent 24 lived at house 250, I wanted them to figure out their way home based on logic, so they knew which direction to head if they were near house 240. I did not code this into the model as I did not know where to start, therefore it does not star in the model, but would be a useful addition to make it more realistic.

When an agent arrives home, the model prints “What a journey! WHERE’S MY BED!!” to let the user know that the drunk, sleepy agent arrived home. Unfortunately, I could not get the model to print the agent’s number to show which agent has arrived home. I tried various ways but could not find a solution, therefore the user must count how many times the statement has been said, or wait until the model has finished when all the agents have returned home.

The final unresolved issue I found was in animating the model. I achieved this in Assessment 1, however I could not get it to work for this drunk agent model. This may be due to the model code not including a number of iterations, thus the animation could not update its frames based on them. I attempted to update the animation frames but failed to get it working.