Author: gy19rf (Richard Frimpong)

Student ID: 201377095

Module Title: Programming for Geographical Information Analysis (Core skills)

Module Code: GEOG5990M

Details about the drunk model:

This independent project allowed me to build an Agent-Based Model (ABM) that shows the movement of drunks a pub in finding their way home as well as stopping their movement as they reach their homes.

Some important files were imported to the model to make the drunks able to move from the pub to their respective homes.

These files included;

-First and foremost, the import operator was used to import all files and functions to create an agent based model.

-csv file which enables the drunks to be exchanged and listed on the model.

-drunk.plan.txt was imported to help read the code.

-Agentframework was used in order to establish drunks in the environment which helps in initialising the drunks, return the description of a drunk as a string of a text etc.

-Matplotlib.pyplot was imported to allow plotting of drunks , pubs, houses and make them visible on the screen for any code reader to easily understand what is going on in the model. Matplotlib.animation was imported to get the model animated.

How the Drunk model can be run:

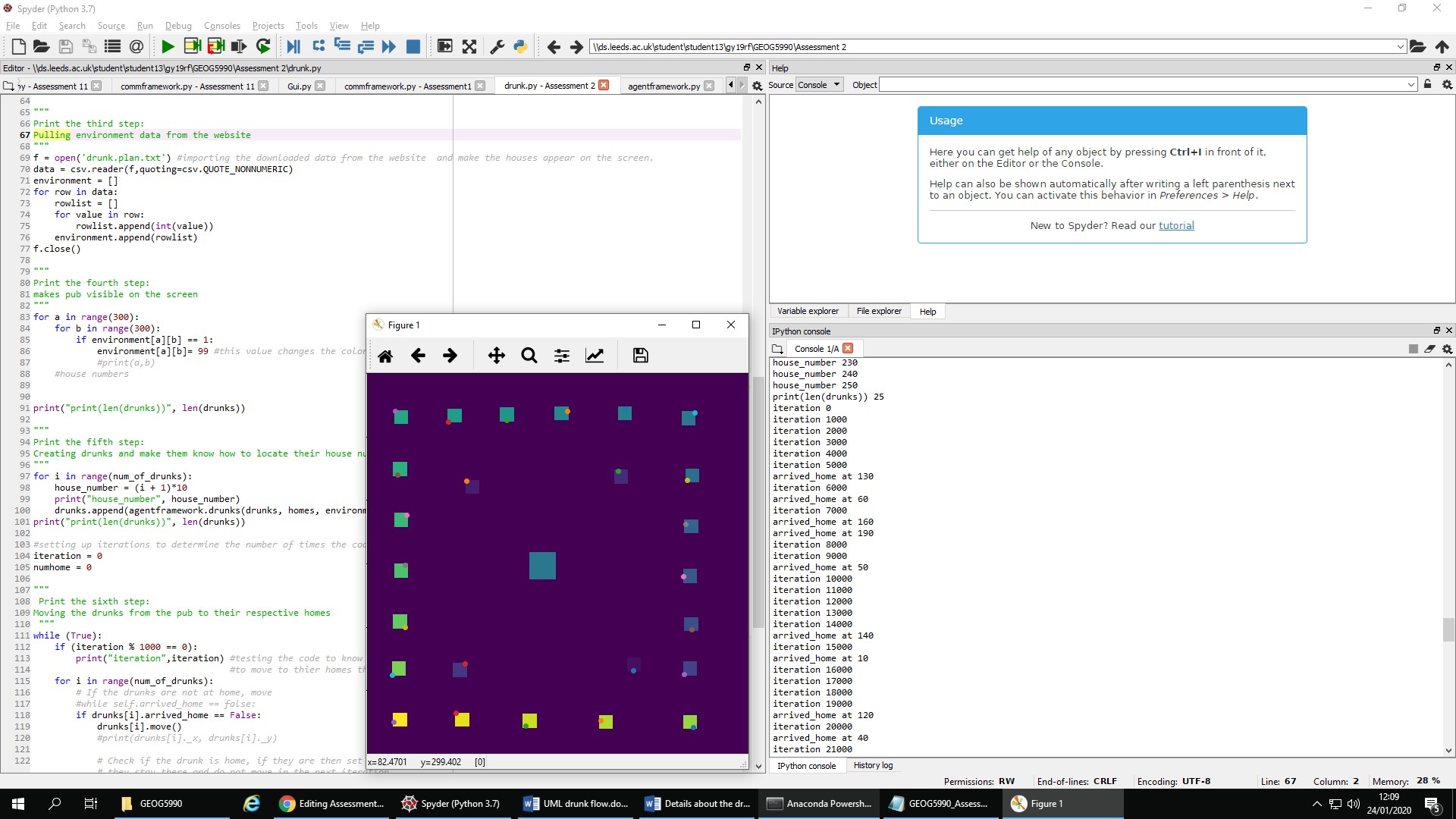
Below are the steps involved in running the Drunk model:

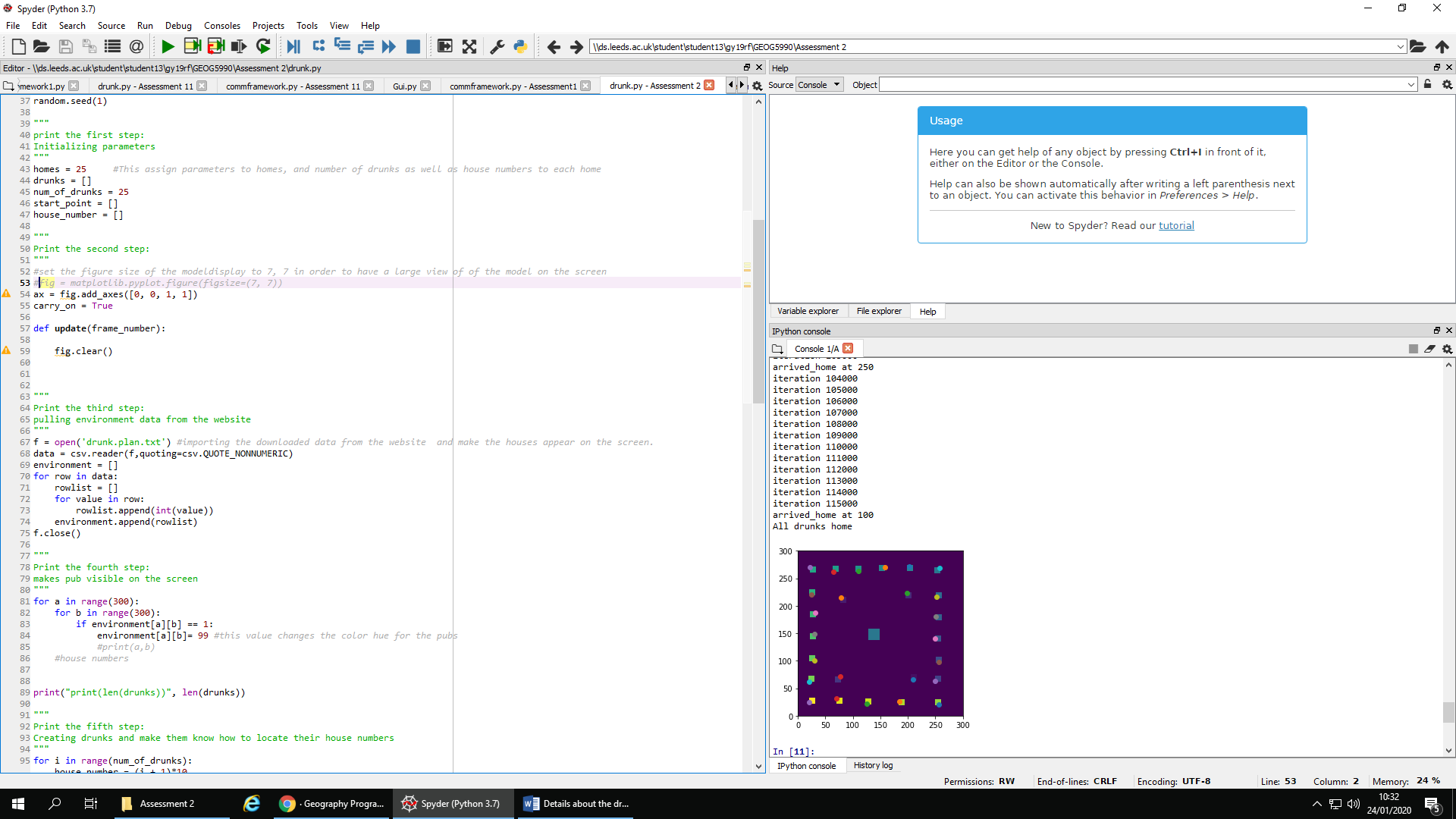
-The Drunk model can be run by clicking on the run file or pressing the F5 button in the anaconda spyder (version 3.7) python software.

What happens after running the model:

After running the code, you will see the number of iterations at which each of the drunks arrived home is being displayed in the Ipython console. Also, the drunk model figure will be displayed, showing that each drunk has successfully reached their home. You will see that all the drunks has successfully arrived home. None of these drunks is expected to be wandering around to find their way home.

See the picture below;





Problem(s) or bugs encounted in this drunk model and the approached used in trying to resolve the bugs:

First and foremost, I realized that the drunks were still moving after setting the drunks to "arrive home". I imported sys.exit from the website and placed it at the buttom of this code;

if environment[drunks[i].\_y][drunks[i].\_x] == drunks[i].house\_number:

drunks[i].arrived\_home = True

print("arrived\_home at", drunks[i].house\_number)

numhome = numhome + 1

if (numhome == num\_of\_drunks):

**sys.exit("All drunks home")**

This made all the drunks to stop moving after reaching their homes.

Again, after making sure that all drunks have stopped moving, I realized that the 15th drunk is still at the pub. I saw that the colour of the pub and that of the house of the drunk are the same. I also thought that the house of the drunk and the pub could have similar x and y coordinate numbers. This may make the drunk to feel that he is already at home.

Below is how I convinced the 15th drunk to move from the pub to his house:

The reason why the 15th drunk refused to move to his home is because, I assigned 150 to the environment when making the pub visible on the model screen. This line of code below made the drunk to stay at the pub even after asking him to move to his home.

for a in range(300):

for b in range(300):

if environment[a][b] == 1:

environment[a][b]= **150**

I therefore changed the environment setting to 99. This made the drunk to move from the pub to his home. Below is the line of code used;

for a in range(300):

for b in range(300):

if environment[a][b] == 1:

environment[a][b]= **99**

The movement of drunks from the pub to their homes were tested using this line of code:

while (True):

if (iteration % 1000 == 0):

print("iteration",iteration) #testing the code to know how the drunks were able

#to move to their

homes through a number of iterations

for i in range(num\_of\_drunks):

# If the drunks are not at home, move

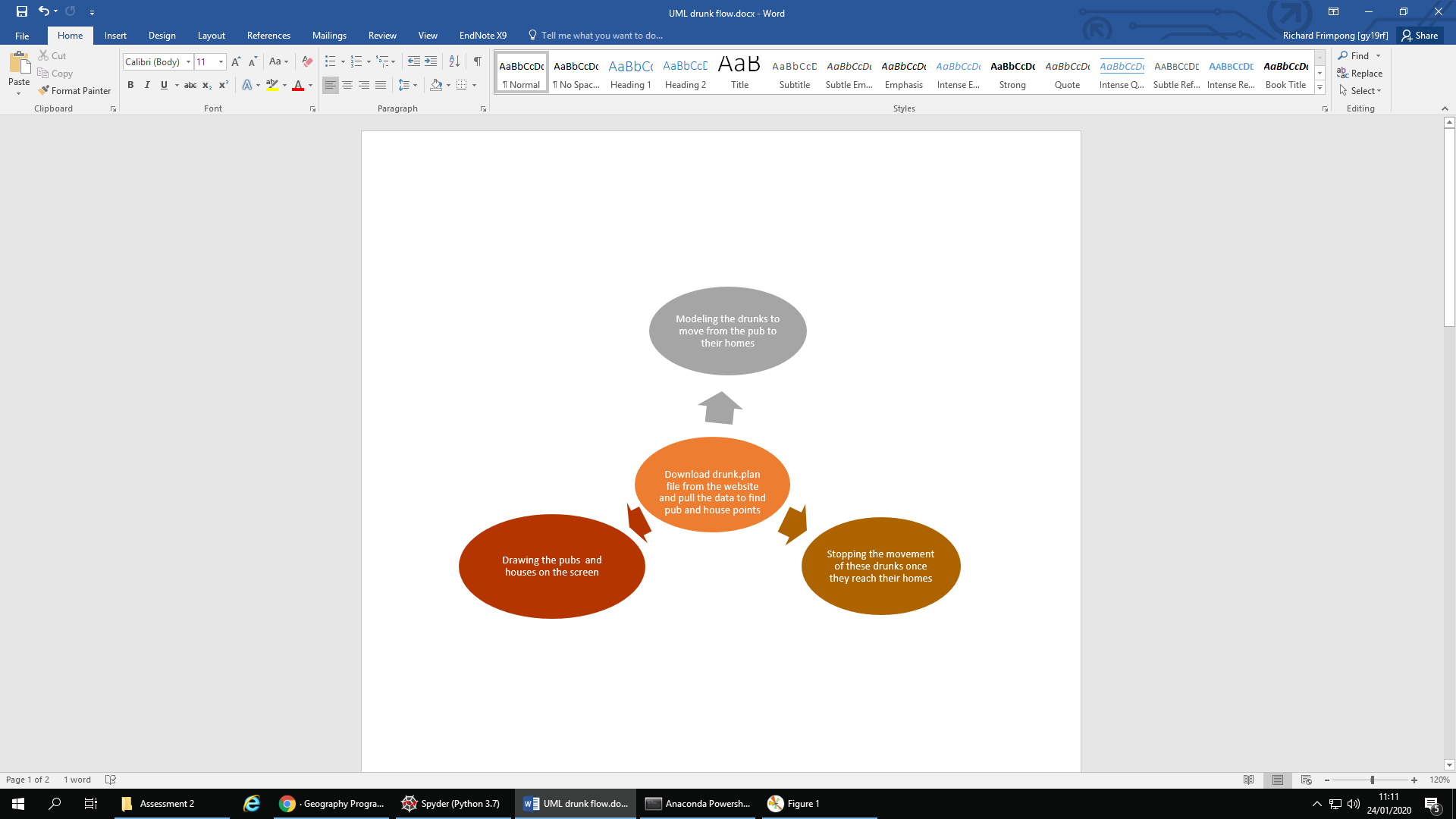
#while self.arrived\_home == false:

if drunks[i].arrived\_home == False:

runks[i].move()

 UML diagram showing either the code structure

Drunk model UML diagram showing code structure:



License: This drunk model has a licensing document attached, which explains the terms and conditions for copying, distribution and modification of this drunk model.