Robbie Henderson – H00011358 Foundations 2 – Assignment Part 2

For the second part of the assignment, due to the first part not fully working, I decided a fresh start would be best, and after hearing of many good results in python, decided to learn python and complete the assignment from now in this language.

I managed to code a simple version of part 1 in python quite quickly and when I moved on to part 2 the biggest difficulty was trying to find documentation on loading files into python. There seemed to be very little material available and after some trial and error I figured out that data is loaded into a dictionary of other dictionaries and lists. Once I had the grasp of this concept the parsing of the file was not too difficult.

The first part of each expression was always an equals followed by a variable, then an argument. I had a function parse_json deal with each expression in the statement list, and create a new entry in my variable dictionary using the variable name and the value in the arguments. The variable would then be output to the output.txt file. If the value was a complex expression I would then pass the expression to a more intelligent function deal_with_new_node, which could handle all types of operators. This function could be called recursively and could handle set creation, tuple creation, equality and membership. Because of the recursive ability, and the fact it only returned what it created, rather than save anything, a set of a tuple and an int could be created as the set function would call the tuple function, save the return and then save the int both in the set.

This also worked well for equivalence and membership as the expressions would be passed back to the top level for comparison, so a set or tuple could be created temporarily for comparison without saving said set or tuple. The code is more thoroughly explained in comments in the code stored in foundations2.py.

Fun Extension -
The program will output an error and create an empty tuple if a tuple of size 1 is attempted.
I decided not to throw an exception as halting the program seemed excessive for such a simple problem. The error is printed in the tuple in the output file and in the terminal

Print -
Printing is handled by recursively reading though each set or tuple, and reading through each set or tuple in the set or tuple, until the bottom level is reached. Each integer is printed, and the relevant brackets for each data structure are printed on the outsides of the values.

Program does not yet test for relationships between alien invasions of the Edwin Chadwick building and the giant rabbit appearances. Sorry. Perhaps in part 3?