Note: I had to change my data set due to the fact that the when trying to read the zip file I would only get gibberish, nothing like what the text said I should be getting where it shows the user ID and friend IDs. When I tried to uncompress it to see issues, I would only get errors, so I changed the data set to the following: https://snap.stanford.edu/data/ego-Facebook.html, with the same question of 6 degrees of separation.

The dataset represents undirected data, and the site also provided some useful information such as the longest shortest path being 8. I now know going into this that my 6 degrees of separation should not be larger than 8 ever, and my average is likely to be much less. I can now use this for tests, just to verify if there is an error with my code.

I start by reading the file and putting it into a data file that I know I can use while error checking, ensuring that issues such as 3 is connected to 3 is not there. After cleaning the data, I want to put the data into a hashmap or a vector so I know what node touches another node. Using the code from lecture 28 on BFS, I created an undirected graph for my data. Since I properly understand the code, as demonstrated in the comments of my code, and I have made adjustments to the code to fit my specific needs, I think it is okay for me to use what the instructor has given. I removed many of the print statements since it is a large collection of data. I instead placed the second value, the one that tells me the distance from one node to the next, into a different value to be added and later calculated for the average distance of separation.

I created a separate module for cleaning/reading the file since it makes it easier to read the code as you don’t have to scroll through data cleaning/reading to figure out what the code is doing. I then ran tests for the number of components, the longest shortest distance and the number of edges. Since I already knew what these values should be, 1, 8 and 88234 respectively, I could test them directly and they all passed. The program takes O(n) time to run since it is a BFS program. After completing the program and running tests to ensure it is correct, I ended up finding that there was an average distance of 3.69 between any two random nodes, well below the 6 degrees of separation.