**Urvi Sharma**

**13 December 2023**

# 210 Project Report

## What does the project do?

The data I chose to use is a Twitter social network set. This dataset has 81306 nodes and 1768149 edges making it reasonably sized. It has edges which represent the users' ID. The project has four main steps: read the data, create the data into a graph, use breadth first search on the graph to analyze connectivity, and implement all of the functions through the main function. It has 3 modules for clarity: graphing, reading, and the main file. The module called reading.rs reads the text file called 'twitter_combined.txt' (located in the src) from a public function called read_data_from_file. The next module is graph.rs. This file creates a graph by making a struct called TwitterGraph. There are then 4 functions under the implementation of Twitter Graph. The first function is add_edge which creates edges between the users in the graph. The next function is min_distance. Min_distance uses breadth first search to find the shortest path between two specified users in the graph. It starts from a particular vertex and visits its neighbors and the neighbors of that neighbor until all of the nodes in the graph are visited. Max_distance is the other function; it is similar to the min_distance function in that it uses breadth first search, but it finds the longest route between the two specified user IDs. The last function is called average_distance_of_user. This function also uses bfs to explore the distance from that specified ID to all the other IDs in the graph. It then takes the average of the distances which gives an idea of how interconnected the ID  is relative to the other IDs. The shorter the average distance means it's more connected to the other users. All of these functions are then called in the main which summarizes what was created. In the main, there is one more function where the user can interact with the code by putting in which IDs they are interested in finding the connection between. This function is called get_user_input. It uses a loop to prompt the user to input the IDs. Once the code is run and the inputs are received, it will output the shortest path (min distance), longest path (max distance), and the average distance of another specified ID.

## How do you run it?

The code is interactive, and once it is run, will prompt the user to input 2 user IDs at first. Once it finds the min and max distance of the two IDs, it will prompt the user one more time to enter one ID. It will then find the average distance from that ID to all of the other IDs in the graph. The user can find that IDs are available in the twitter_combined.txt file which is in the src. I have also provided some possible IDs that can be used in the program here:

178852084  228078044

197668807  234047967

86799233  57730647

61693492  271602109

519072110  44473103

185787222  281707984

7398912  276597679

93579404  104585489

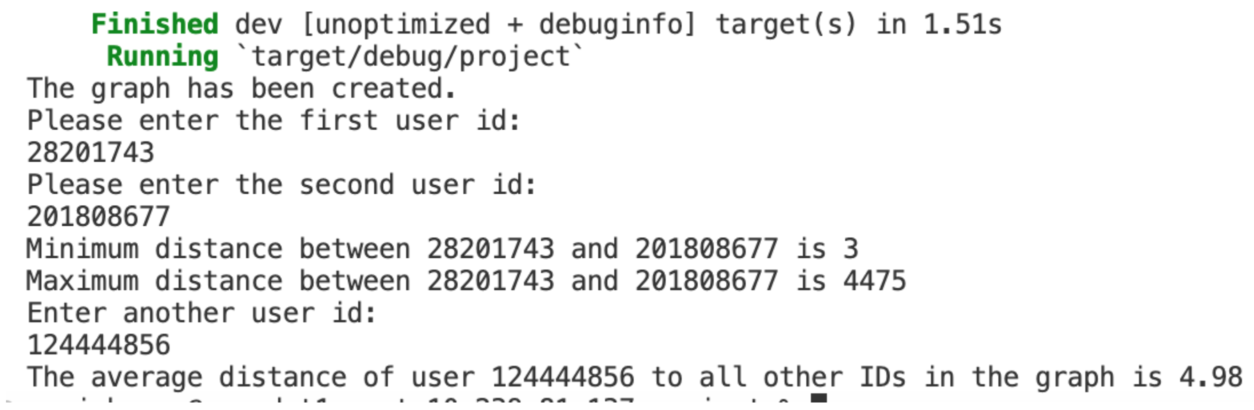
79721137  49636316

8362812  17132311

Any of these IDs can be used in any order.

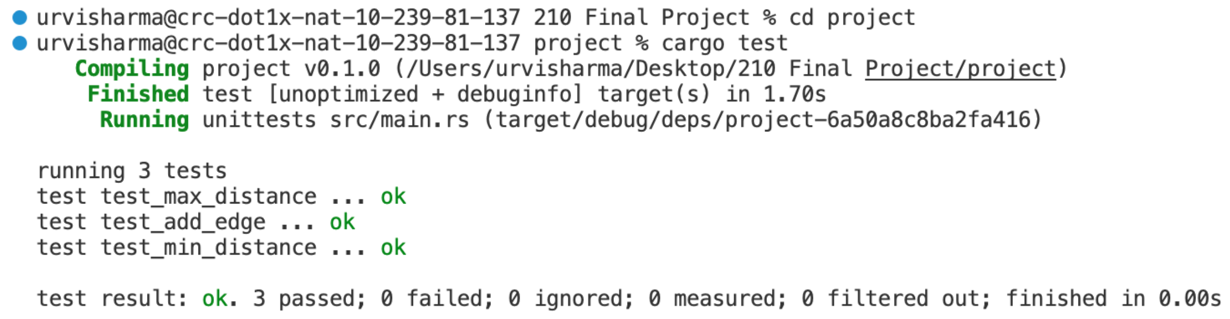
## What does the output look like?

I have entered 28201743 as my first user ID. Then I entered 201808677 as my second user ID. My program found the shortest and longest path length between those IDs. I was then prompted to enter another user ID, so I chose ID 124444856. My program then displayed the average distance of that user relative to all of the other IDs in the graph.



```
    Finished dev [unoptimized + debuginfo] target(s) in 1.51s
     Running `target/debug/project`
The graph has been created.
Please enter the first user id:
28201743
Please enter the second user id:
201808677
Minimum distance between 28201743 and 201808677 is 3
Maximum distance between 28201743 and 201808677 is 4475
Enter another user id:
124444856
The average distance of user 124444856 to all other IDs in the graph is 4.98
```

## Tests:



```
● urvisharma@crc-dot1x-nat-10-239-81-137 210 Final Project % cd project
● urvisharma@crc-dot1x-nat-10-239-81-137 project % cargo test
     Compiling project v0.1.0 (/Users/urvisharma/Desktop/210 Final Project/project)
      Finished test [unoptimized + debuginfo] target(s) in 1.70s
       Running unittests src/main.rs (target/debug/deps/project-6a50a8c8ba2fa416)

  running 3 tests
  test test_max_distance ... ok
  test test_add_edge ... ok
  test test_min_distance ... ok

  test result: ok. 3 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

All of my 3 tests passed.