**CS 32 Project 4 Brief**

I finished the whole project. I did not find any bugs. I did not use any banned STL components.

Testing Process:

1. I initially tested the User, Movie, and treemm on a subset of data - just 4 datapoints. I ensured that it works properly. I won’t mention those test examples as they are trivial and tests on UserDatabase and MovieDatabase also test User, Movie, and treemm.
2. UserDatabase:
3. Testing load: Loaded users.txt multiple times to check if it returns false after the first load and does not affect the initial load.
4. Tested all methods on valid emails such as [BraylonNewt899@cox.net](mailto:BraylonNewt899@cox.net) and [AbelS2@zoho.com](mailto:AbelS2@zoho.com). I also tested the first and last emails.
5. I tested invalid emails by changing the case (to check case sensitivity), such as [bRAylonNewt899@cox.net](mailto:bRAylonNewt899@cox.net). It resulted in bad access error which is expected behavior (because it returns a nullptr).
6. MovieDatabase:
   1. Testing load: Loaded movies.txt multiple times to check if it returns false after the first load and does not affect the initial load.
   2. Tested get\_movie\_from\_id with randomly picked out IDs.
   3. Tested rest of the methods outside load with the names of famous directors, actors, and genres. For instance, I tried out Steven Spielberg, Christopher Nolan, Tom Cruise, Romance, etc. Then I verified that the results are correct with the data. Tested with different casing (to test case insensitiveness).
   4. Using invalid movieIDs as test case caused a bad access error as expected. Using invalid directors, actors, and genres returned empty container.
7. Recommender:
   1. Used the find\_match function given in the spec to do all testing.
   2. Tried finding recommendations for invalid emails – (I ensured that this returns an empty vector)
   3. Created a smaller dataset of 10 movies and used [BraylonNewt899@cox.net](mailto:BraylonNewt899@cox.net) to get recommendations. Calculated scores by hand and checked if it matched the ordering recommendations given and compatibility scores. Also ensured that the user watch history did not have any movies common to the recommendations. Conducted similar testing on [HezekF0394@aol.com](mailto:HezekF0394@aol.com) too.
   4. Manipulated email casing to ensure that the recommendation fails and tried asking for more recommendations than the number of movies we have to ensure that we get only movies which have ratings of 1 and above.
   5. Conducted similar tests on the bigger dataset using random emails and compared the results with classmates. [LBla4683@cox.net](mailto:LBla4683@cox.net) was a particularly helpful test case because the user just watched 3 movies.
   6. Tested everything for memory leaks on g32.