Richard Bustamante

CSC 415 Fall 2017

Assignment 1 – Application Development

readme.pdf

Full link to Project on VM: student1@csc415-server05.hpc.tcnj.edu

Instructions:

After running the program the User will be prompted with a Menu that displays 9 options for the User to choose from. The user is expected to enter in a number that corresponds to their selection in the Menu. If the User inserts a character that is not included in the possible Menu options, it will be caught and the User will be given the opportunity to input again. If a User wants to leave the Menu there is a "Quit" option which is number 9.

If the User chooses to input a filename to be added as a Set, they will choose 1, which will prompt the User for a filename and the program will read the file and put the file into a "Set" which I have implemented with the use of a Hash data type. Each file's information will be placed into a Hash with a Hashname that is the same as the filename. I chose to do it this way because no files can be in the same directory if they have the same name, thus making every Hash unique. If the User inputs a file that is not found, it will be caught and dealt with accordingly.

If the User chooses to add an element to an existing Set, they will be prompted for the filename that they would like to add an element to, if this file has already been added as a Set using option number 1 in the Menu. They will then be prompted to specify the filename, the account number, and the name associated with the account number and both will be added to the end of the specified Set.

Option 3 allows the User to find the Union of two Sets that have already been added. The User will be prompted based on how many Sets have been added thus far. They will have the choice of which two Sets to find the Union of and will display the result.

Option 4 allows the User to find the intersection of two Sets that have already been added using option 1. The User will then be prompted based on how many Sets have been added thus far. They will have the choice of which two Sets to find the intersection of and will display the result.

Option 5 allows the User to find the Cartesian Product of two Sets that have already been added via option 1. The User will be prompted based on how many Sets have been added thus far. They will have the choice which two Sets to find the Cartesian Product of and will display the result.

Option 6 allows the User to display any Set, that has already been added via option 1. They merely need to specify which filename they would like to display.

Option 7 and 8 work the same way. They ask the User for which file they would like to find if it is Empty or Full, respectively. The User will be prompted for the filename and the result will be displayed.

Known bugs, issues, and limitations

Some known limitations of this program are its efficiency. For larger files the process of adding the large file to a Set may be lengthy since it works sequentially, reading each line by line and adding each line to the Subscriber class and then take each like and add it to its own respective Hash. Some other limitations include being incapable of running an intersection, Cartesian product, or union after having ran one of the other operations. For some reason Ruby does not allow for me to access the filenames more than once if I try to read in from a previously used filename it breaks. Another known error is if the User inputs filenames that are the same, as mentioned before, in a directory filenames are unique but if the User does decide to try and input the same file more than once the program breaks.