**Brief Description:**

The design is set to start with the bank file and will read the transactions from the text file and put them into a queue to be used for the actual bank process(the execution function). The execution function will then read in from the queue and checks to see what type of transaction it is and execute it according to the type. When a new account is added, it will be sent to be added to the BSTree and be sorted into the tree. I'm setting it up so that when the transactions are being executed, it will process the actual transaction and then sends the final result to the setter transaction in the account class depending on what type of transaction it was.

The BSTree has its own class and will be called in the Bank class in order to keep the account ID's saved. The Account class is connection to the bank class through the transaction types, because after the transaction is done, the result will be sent to the transaction setter in the account class using the parameters fund and amount.

**STEPS:**

* Read in from file
* Transfer all transactions to a queue
* Execute the transactions
* For the open function: if the account ID exists, send error saying that ID exists.

Else: create the new account and add it to the BSTree

* For the other transaction types: If the ID exists, continue with the transaction.

Else: Output saying that the ID does not exist.

* Before the transactions finish, the account class setter will take the result of the transaction and the fund type as a parameter in order to set up the account's current balance in that specific fund
* After all transactions are finished, display the history of all account ID's and their new balance per fund.
* The getter of the account class will help display the history of each account.