Program 5 – Kranti Paudyal

The Jolly Banker

For this program I have five classes that work together to accomplish the goal of the program. The classes are:

* Account: this class is responsible for one of the core functions of the program which is the flow of money, creating an account and storing 10 different funds within the account. This class will be able to make deposit to the account, transfer from one account to another, withdraw from the account and open an account. This class works with fund class to keep track of money as well as all the transactions that occurred during the execution of the program.
* Fund: Fund class keeps track of money and history of the funds, as instructed by the account class, this class can handle adding amount to fund, deducting amount from fund, and keeping track of all the funds
* BinarySearchTree: Binary search tree class is corely dependent on the data structure BST. It is responsible can store accounts of the whole program, it can retrieve the account from the tree, check if account already exists, and display the whole tree which contains all the accounts that is hold by the bank application.
* Transaction: Transaction class plays a vital role on making different actions based on the input from the text file. This class creates transaction objects which needs to be processed from the input
* Bank: Bank class has a big role on reading file input, creating a queue of transactions and executing the transactions. This class reads in the .txt file and inserts all the actions in the queue in orderly manner, this class works with Binarysearchtree class to insert, find, retrieve accounts for executing transactions, and with transaction class to make the input a order that the program can follow. At the end, this program can print out all the actions that were performed in the execution period.

Class diagram:

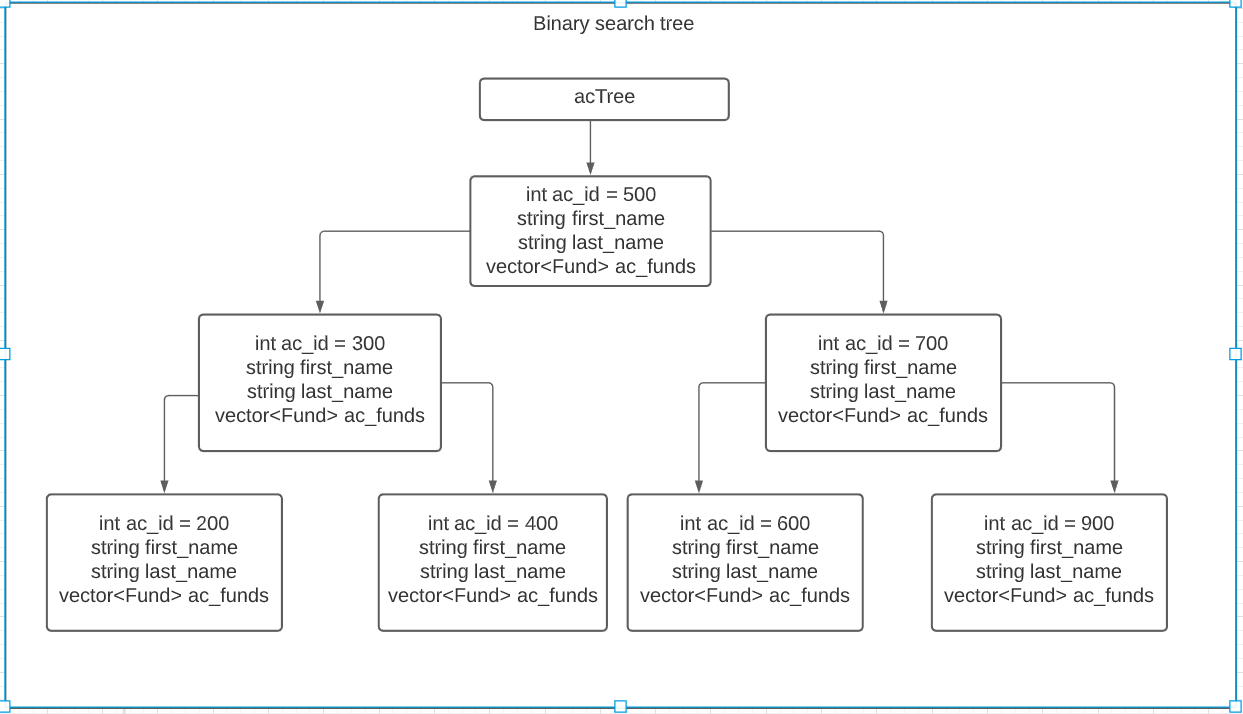
Diagram

Description automatically generated

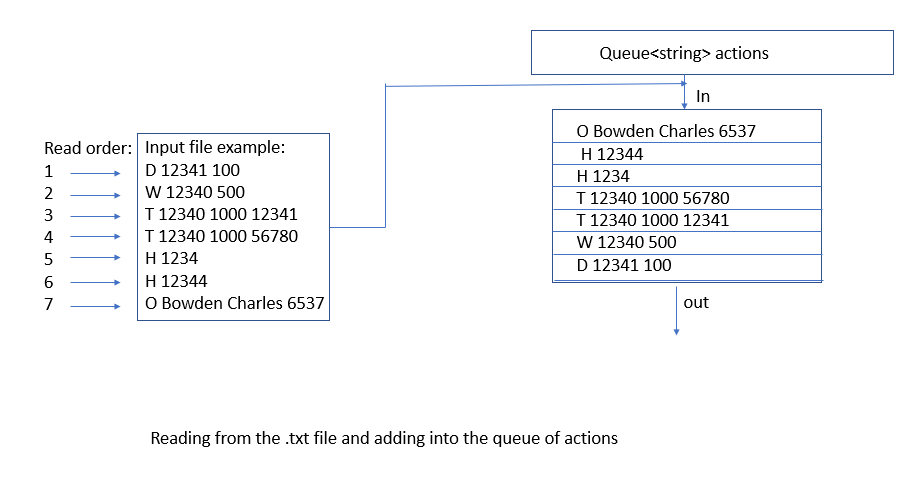
This is high level design of how classes interact

Visuals of key data structures:

* Vector to keep track of funds and history of transactions
* Queue to store input as list of transactions
* Binary search tree to store and work with the accounts



Reading into the Queue:



Diagram

Description automatically generated

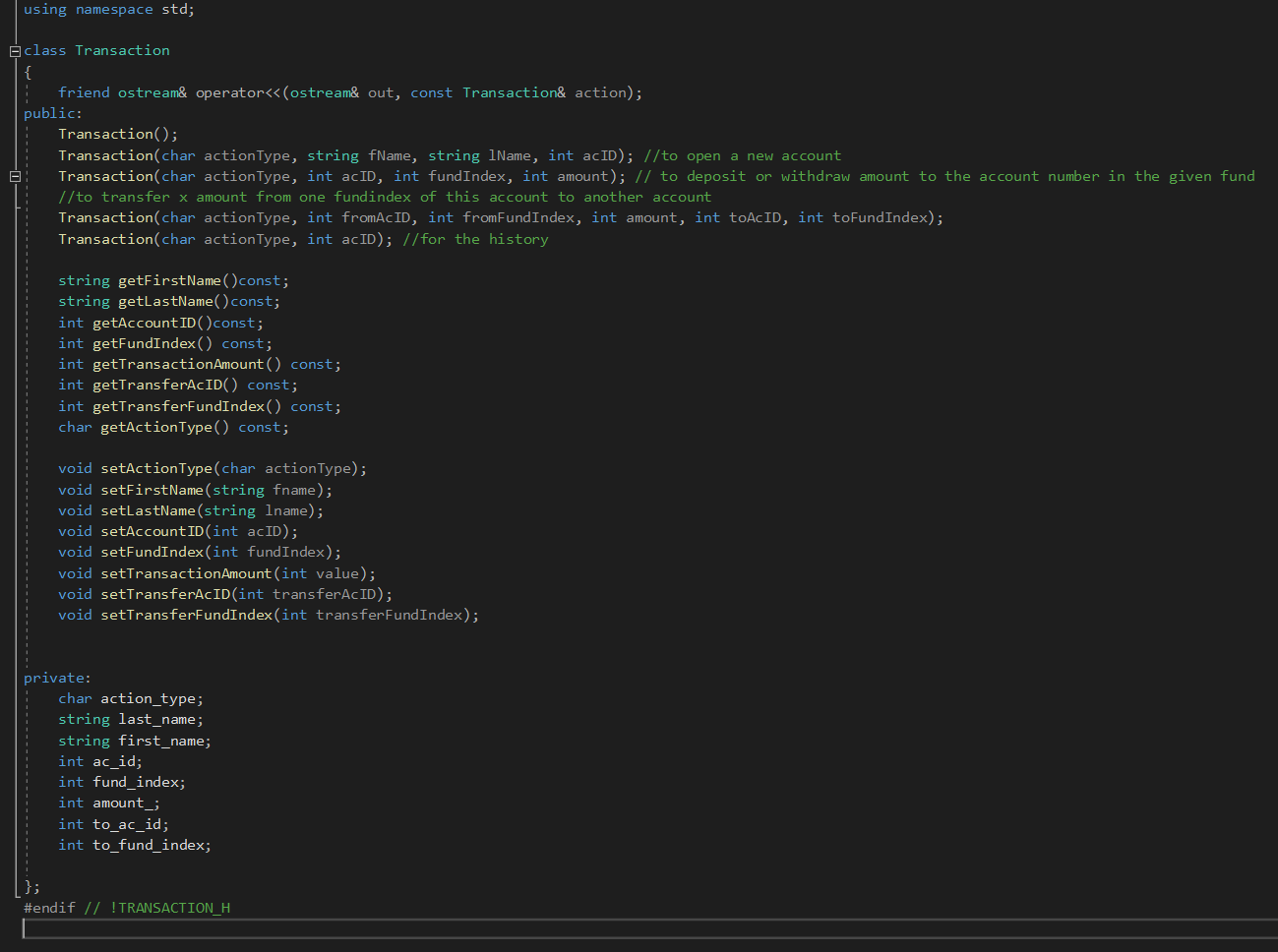
Flow of program Mostly visual in the class diagram

Text

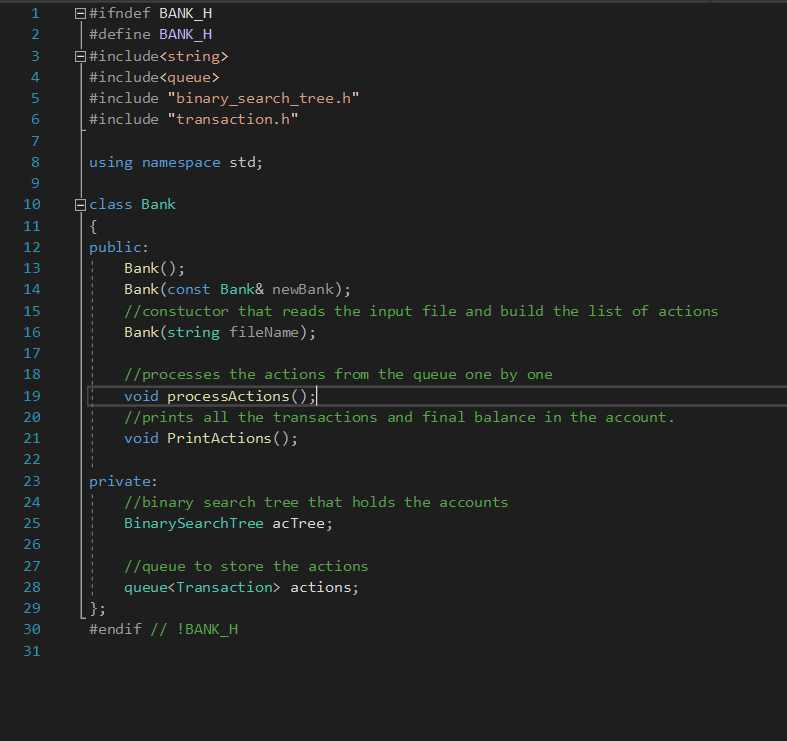
Description automatically generatedBinarySearchTree.h

Account.h

  
 ewafsdfas

Transaction.h

Bank.h



Fund.h

