Brandon Piotrowski

COMP-SCI-303

Final Project Report

**Assumptions**

The main two assumptions I had while creating the project was adding an interface for the user of the simulation program to manually input each customers information in the order they arrived at the bank and I also assumed that the transaction type and transaction amount would be stored within the Customer object instead of storing the customer and the transaction as a pair in the stack. To do that I had the customer class inherit the transaction class. Some other, smaller assumptions were that there needed to be error handling and input validation, that the output of the stack included the customer’s current account balance, transaction type, and transaction amount, and I also assumed that when manually inputting the customer’s the system would not allow a customer to overdraw from their account.

**UML Diagrams**

**Diagram

Description automatically generated**

**Big-O**

**Function: readCustomer(stack<Customer> & customers, string fileName)**

This function is O(n2). This is because there is an outer while loop that runs while the input file stream is still able to read or n times. Then there is an inner while loop that runs until the stringstream can no longer read any items from the current line.

**Function: manualInputCustomers(stack<Customer>& customers)**

This function is O(n2) because there is an outer while loop that runs while the user still wants to add customers so it executes n times. Then there are multiple separate while loops that also run n times while the user inputs an invalid input. So the whole functions worst case scenario is O(n2).

**Function: simulateBankTeller(stack<Customer>& customers)**

This function is O(n) because there is one while loop that runs n times until the stack of customers is empty.

**References**

No references were used other than the powerpoints available in the class modules.