**Design Document**

Introduction

This program is a bank simulation, where a priority queue tracks customer arrivals and departures based on a text file listing their arrival and transaction times. The program's goal is to, in order, list arrivals and departures, count the total number of customers, and calculate their average waiting time. This simulation processes events in the order they occur, as indicated by the arrival times, and it also provides insight into the efficiency of the bank's customer service.

Data Structures

The program uses two data structures: queues and priority queues. The priority queue, implemented using a link-based approach, efficiently organizes customer arrivals and departures by tracking them from an input file. A standard queue acts as the waiting line at the bank, handling customers in the order of their arrival. In the provided code, the SL\_PriorityQueue manages events by their scheduled times, ensuring that events with the earliest times are processed first. The LinkedQueue mimics the real-life queue of customers at a bank, holding each new arrival until the teller is available. This setup allows for an accurate representation of a bank's customer service flow, emphasizing the practical application of these data structures in managing and processing real-time events.

Functions

This program has 2 functions.

* The arrivalEvent function processes bank customer arrival events. When a customer arrives, the function first removes this event from the eventLog priority queue.
* If the bank line is empty and the teller is available, the function calculates the customer's departure time, creates a departure event, adds it to the event log, and updates the teller's status and total wait time.
* If the bank line is not empty or the teller is busy, the arriving customer's event is added to the bankLine queue, indicating they must wait.
* The departEvent function manages the departure of customers from the bank. It starts by removing the current departure event from the eventLog priority queue.
* If there are customers waiting in line (bankLine is not empty), the function processes the next customer: it calculates their departure time, updates the total wait time, and modifies their event to reflect departure, then adds this updated event to the event log.
* If there are no customers waiting, the function simply sets the teller's status to available (true), indicating they are ready for the next customer.
* The event file has the typical get and set functions for its data members.
* It also has overloaded functions for >, <, ==, !=, >>, <<.
  + For less than and greater than they are revered because we want the arrival time priority to be less.

The main program

The main program simulates a bank environment, tracking customer arrivals and departures using a priority queue. It reads customer arrival and transaction times from a text file, creating events for each customer. These events are managed in a priority queue (eventLog) to ensure they're processed in the correct order. The program distinguishes between arrival ('A') and departure ('D') events, handling them using arrivalEvent and departEvent functions. The arrivalEvent function adds customers to a queue (bankLine) if the teller is busy, or schedules their departure if the teller is free. Finally, departEvent processes customer departures, updating the teller's status and calculating total and average wait times, with the program outputting these statistics at the end.

**Structure of the main program**

File of text Main program

SL\_PriorityQueue LinkedQueue

LinkedSortedList PrecondViolatedExcep PrecondViolatedExcep Node

PriorityQueueInterface QueueInterface

**User Document**

This program is a bank simulation, a priority queue tracks customer arrivals and departures based on a text file listing their arrival and transaction times. The program's goal is to list arrivals and departures, count the total number of customers and calculate their average waiting time. This simulation processes events in the order they occur, as indicated by the arrival times.

The program’s name is BankPQSystem.cpp. It is in the following directory on CentOS.

home/STCLOUDSTATE/it1032ao/CSCI301/Project7/BankSystem

To compile it enter:

g++ BankPQSystem.cpp

To run it then enter:

./a.out

An example run of the program would look like this:

A screenshot of a computer

Description automatically generated

Have an input file of arrival and transaction lengths.

Then run the program as instructed above. You then will get the following output in the console:

A screen shot of a computer

Description automatically generated

Note: other formatted input files will not work.