Bank Simulation Application User Document

The Bank Simulation Application is a C++ program that simulates the operation of a bank by modeling the arrival and departure of customers. The simulation is run using an event-driven approach where events such as customer arrivals and departures are modeled as objects of a class named Event.

Installation and Running

To run this application, follow the instructions below:

Ensure that you have a C++ compiler installed on your computer.

Download all the necessary files including "BankSimApp.cpp" and "Event.h".

Compile the application

Run the compiled application by executing the command: ./BankSimApp

Usage

Upon running the application, you will be prompted to input the arrival time and transaction time of each customer. The program will simulate the arrival and departure of customers based on these inputs. After the simulation is complete, the program will output the final statistics including the total number of people processed and the average amount of time spent waiting.

Code Structure

The Bank Simulation Application consists of two main files: "BankSimApp.cpp" and "Event.h".

BankSimApp.cpp

This file contains the main program logic including the simulation function simulate(), arrival and departure event processing functions processArrival() and processDeparture(), and the main() function.

The simulate() function initializes the event list and bank queue, processes arrival events and departure events until the event list is empty, and outputs the final statistics of the simulation.

The processArrival() function processes an arrival event by either serving the customer immediately if the teller is available or by adding the customer to the bank queue.

The processDeparture() function processes a departure event by either serving the next customer in the bank queue or by setting the teller to available if the queue is empty.

Event.h

This file contains the definition of the Event class used to represent arrival and departure events. The class contains member variables arrival, time, and transactionTime, and member functions isArrival(), getTime(), and getTransactionTime().

Conclusion

The Bank Simulation Application is a useful tool for simulating the operation of a bank and analyzing customer traffic. With this user document, you should be able to understand how to install and run the program, as well as how the code is structured and how it works.

Design Document - BankSimApp

Introduction:

The BankSimApp is a simulation program that simulates the waiting time of customers in a bank. It uses a priority queue to keep track of the arrival and departure events of the customers. The program is written in C++.

Program Flow:

The program begins by creating an empty priority queue and an empty bank queue to represent the event list and bank line, respectively. Then, it reads input from the user which includes arrival time and transaction time of each customer, and creates an arrival event for each customer, which is added to the priority queue.

The program uses an event loop to process events from the priority queue. The loop runs until the priority queue is empty. If the next event is an arrival event, the program processes the event by either sending the customer to the teller or adding the customer to the bank queue, depending on the teller availability. If the next event is a departure event, the program processes the event by either sending the next customer in line to the teller or setting the teller as available, depending on the availability of the bank queue.

The program keeps track of the number of people processed and the total waiting time, and calculates the average waiting time after the event loop ends. The program then outputs the final statistics.

Data Structures:

Priority Queue: The priority queue is used to keep track of the events. The priority queue is sorted based on the time of the events, and in case of a tie, the arrival event is given higher priority than the departure event.

Queue: The queue is used to represent the bank line. The queue is a first-in-first-out data structure, and it is used to store customers who are waiting in line.

Functions:

void simulate(): This function is the main function of the program. It creates an empty priority queue and an empty bank queue, and reads input from the user. It then enters an event loop to process events from the priority queue, and calculates and outputs the final statistics.

void processArrival(Event arrivalEvent, PQueue& eventListPQueue, Queue& bankQueue): This function processes an arrival event. It creates a customer event from the arrival event and either sends the customer to the teller or adds the customer to the bank queue, depending on the teller availability.

void processDeparture(Event departureEvent, PQueue& eventListPQueue, Queue& bankQueue): This function processes a departure event. It either sends the next customer in line to the teller or sets the teller as available, depending on the availability of the bank queue.

bool PQueue::compare(const Event& element1, const Event& element2) const: This function compares two events in order to sort them correctly in the priority queue.

Classes:

Event: The Event class represents an event in the bank simulation. It has three private member variables: arrival, time, and transactionTime. The arrival variable is a boolean variable that indicates whether the event is an arrival event or a departure event. The time variable represents the time of the event, and the transactionTime variable represents the transaction time of the customer.

PQueue: The PQueue class represents the priority queue used in the bank simulation. It has one private member variable: head, which is a pointer to the head of the priority queue. It has several public member functions, including a constructor, a destructor, and functions to check if the priority queue is empty, enqueue an element in the priority queue, dequeue an element from the priority queue, and compare two events in order to sort them correctly in the priority queue.