Experiment topic: Design and Realization of Banking Business Simulation System

1. Experimental content

Design and Implementation of Project 3 Banking Business Simulation System (8 credit hours)

1. Description of the problem: Assume that a bank has four windows to receive customers from the outside world, and customers have been entering the bank since the bank opened in the morning. Since each window can only receive one customer at a certain time, it is necessary to line up in front of each window in turn when there are a large number of customers. On the contrary, if all four windows are occupied by customers, he will be behind the line with the least number of people. Now it is necessary to write a program to simulate this business activity of the bank and calculate the average time that customers stay in the bank in a day.

2. basic requirements

(1) Initialize (OpenForDay) to simulate the state of each data structure when the bank opens.

(2) Event-driven (EventDrived) , to deal with customer arrival and departure events accordingly.

(3) Off-duty processing (CloseForDay), simulate the action when the bank is closed, and count the average stay time of customers.

(4) Design independently according to the requirements of the topic, and write a design report as required after the design is completed .

2.Design ideas

1、initialization

（1）The queue is mainly used to simulate the queue situation in front of the bank window. Design the structure variable of the queue element, which contains <user serial number, time when the user enters the bank, and time required for separate processing of the user's business>

2、event driven

（1）call #include <time.h>

The value is randomly generated according to the time difference between users entering the bank and the time required for the user's business to be handled separately. Assume that a new user enters the bank every 1-10 minutes, and the transaction time for each user is between 3-30 minutes.

（2）After the user finishes processing, leave the team. The new user selects the team with the shortest captain to join the team.

3. After work

（1）A working day is 480 minutes. When the 480 minutes are full, the customer's stay in the bank is counted and included in the total time. At the same time, the total number of people on the day is recorded to obtain the average stay time.

3. Experiment code

Two versions are given below: (divided into single-file version and multi-file version)

Code (single-file version):

```
#include <stdio.h>
#include <stdlib.h>

#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#define OK 1
#define ERROR 0
#define TRUE 1
#define FALSE 0

typedef struct QElemType{ //queue element
int entime;//Customer enters the bank time
int detime;//The time required to handle business
    int stime;
    int senum;
struct QElemType *next;
}QElemType;

/******************Queue operation******************/
typedef struct QNode
{
    QElemType data;
    struct QNode *next;
}QNode,*QueuePtr;
typedef struct
{
    QueuePtr front;
    QueuePtr rear;
    int stoptime;
    int num;
}LinkQueue;

int InitQueue(LinkQueue &Q)
{
    Q.front=Q.rear=new QNode;
    Q. front->next=NULL;
    Q.num=0;
    return OK;
}
```

```
int EnQueue(LinkQueue &Q, QElemType e)
{
	QueuePtr p=new QNode;
	p->data=e;
	Q.num++;
	p->next=NULL;
	Q. rear->next=p;
	Q. rear=p;
	return OK;
}
int DeQueue(LinkQueue &Q, QElemType &e)
{
	if(Q. front==Q. rear)
		return ERROR;
QueuePtr p;
	p=Q.front->next;
	e=p->data;
	Q.num--;
	Q. front->next=p->next;
	if(Q. rear==p)
		Q. rear = Q. front;
	delete p;
	return OK;
}
int QueueLength(LinkQueue Q){
//Return the length of the queue Q, that is, the number of elements
int count=0;
QNode *p=Q.front->next;
while(p){
p=p->next;
count++;
}
return count;
}
QElemType GetHead(LinkQueue Q) //Get the stime of the queue head element
{
	if(Q. front!=Q. rear)
		return Q. front->next->data;
}
bool EmptyQueue(LinkQueue Q)
{
	if(Q. front==Q. rear)
		return 1;
```

```
    else
        return 0;
}
void show(LinkQueue Q) //Display the information of each element in the team
{
    QNode *p=Q.front->next;
    while(p)
    {
        printf("<%2d,%3d,%2d> ",p->data.senum,p->data.entime,p->data.detime);
        p=p->next;
    }
}
int modifyclose(LinkQueue Q) //Calculate the stay time of unfinished users when closing the door
{
    int a=0;
    QNode *p=Q.front->next;
    while(p)
    {
        a=a+480-p->data.entime;
        p=p->next;
    }
    return a;
}
/********************The main program******************/

int main()
{
    printf("**************Banking Simulation System**************\n");
    printf("System description:\nThe content in the following <> refers to <customer number, the time when the customer enters the bank, and the time required for customer business>\n");
    printf("In this simulation, it is assumed that a new customer enters the bank every 1-10 minutes\nThe bank goes to work from 9:00-17:00 on the same day, a total of 480 minutes\n\n");
    printf("*****************Simulated system startup*****************\n");
    int i,j; //i,j is the number of for loop calls
    int k1=0; //k1 is used to mark event 1: leave
    int k2[2]={0,0}; //k2[0] is used to mark event 2: empty team enters, k2[1] is used to record the window number
    int k3[2]={0,0}; //k3[0] is used to mark event 3: non-empty queue entry, k3[1] is used to record the window number
    LinkQueue Q[5]; //Build a team, use the team to simulate the queuing situation in front of the window
    for(i=1;i<=4;i++)
        InitQueue(Q[i]);
    QElemType e;
```

```
int num=0;int alltime=0; //total number of people, total waiting time
srand(time(NULL));
int interval=rand()%10+1; //The time difference between the customer's continuous access to the bank
for(int timenow=0;timenow<480;timenow++)
{
    k1=0;k2[0]=0;k2[1]=0;k3[0]=0;k3[1]=0;
    int h[5]={0,0,0,0,0};
    for(i=1;i<=4;i++) //If it is time for a customer to leave
    {
        if(EmptyQueue(Q[i])!=1)
        {
            QElemType m;
            m=GetHead(Q[i]);
            if (timenow==m.stime)
            {
                h[i]=i;
                k1++;

            }
        }
    }
    if(timenow==interval) //When a new customer enters the bank
    {
        QElemType a; //Create temporary customer data
        a.entime=timenow;
        a.detime=rand()%28+3; //Each customer takes between 3 and 30 minutes
        interval=interval+rand()%10+1; // Create a new time for the next customer to enter the bank
        int t=0; //t marks whether all 4 teams are empty
        for(i=1;i<=4;i++) //If there is an empty team, send it directly to the front team
        {
            if(EmptyQueue(Q[i])==TRUE)
            {
                if(Q[i].stoptime<timenow)
                    Q[i].stoptime=timenow;
                Q[i].stoptime=Q[i].stoptime+a.detime;
                a.stime=Q[i].stoptime;
                num=num+1;
                a.senum=num;
                k2[1]=num;
                EnQueue(Q[i],a); //Send to team i
k2[0]=i;
                t++; //t marks whether there is an empty team
```