实验二-离散事件的模拟

PB17081504
廖洲洲

1. 实验要求

- 实现银行业务的模拟
- 假设银行有四个窗口，模拟每天客户到达银行和离开银行的过程
- 计算客户在银行的平均逗留时间

2. 实验内容

- 创建事件表，事件的主要信息是事件类型和事件发生的时刻。处理的事件有两类，一类是客户到达事件，另一类是客户离开事件。按事件发生时间的先后顺序进行程序驱动。
- 创建窗口队列。程序中需要 4 个队列，队列中客户的主要信息是客户到达的时刻和客户办理事务所需的时间。银行队列中的队头客户即为正在窗口办理事务的客户,他办完事务离开队列的时刻就是即将发生的客户离开事件的时刻。这就是说，对每个队头客户都存在一个将要驱动的客户离开事件。
- 随机生成顾客到达银行，每次生成顾客办理业务时间和下一个顾客到达的间隔时间。假设每个客户办理业务的时间不超过 30 分钟，两个相邻到达银行的客户的时间间隔不超过 5 分钟
- 每次到达的客户将在队伍最短的窗口排队。
- 每一个事件完成后即驱动下一个事件发生

3. 实验关键代码讲述

事件的驱动，只要事件表不空，则取出该次发生的事件

```
int main()
{
    printf("请输入银行模拟时间：\n");
    scanf("%d",&CloseTime);
    //银行排队模拟
    OpenForDay();
    srand((unsigned)time(NULL));
    while(!ListEmpty(&ev)){
        DelFirst(&ev,&en);
        printf("**********事件**********\n");
        if(en.NType==0)
            CustomerArrived();
        else
            CustomerDepature();
        PrintQueue();
        PrintEventList();
    }
    printf("\n顾客逗留总时间总时间：%d min,顾客平均逗留时间：%g min.\n",TotalTime,(float)TotalTime/CustomerNum);
    system("pause");
}
```

生成随机数，即顾客到达，然后将其插入最短队列，同时将下一人的到来插入事件表

```
void CustomerArrived(){
    //顾客到达事件
    int durtime,intertime,i,t;
    QElemType e;
    CustomerNum++;
    intertime=rand()%5+1;       //间隔时间在5分钟内
    durtime=rand()%30+1;        //办理业务时间在30分钟内
    t=en.OccurTime+intertime;
    if( en.OccurTime<CloseTime){          //银行尚未关门
        printf("顾客到达时间:%d,该顾客办理业务所需时间:%d,下一位顾客到达时间间隔:%d|\n",en.OccurTime,durtime,intertime);//
        OrderInsert(&ev,NewEvent(t,0));
        i=ShortestQueue();          //最短队列
        e.ArriveTime=en.OccurTime;
        e.Duration=durtime;
        EnQueue(&q[i],e);
        if(QueueLength(q[i])==1)
            OrderInsert(&ev,NewEvent(en.OccurTime+durtime,i));
    }else{
        printf("一位顾客到达:%d，但银行不再接受业务办理了！\n",en.OccurTime);
        CustomerNum--;
    }
}
```

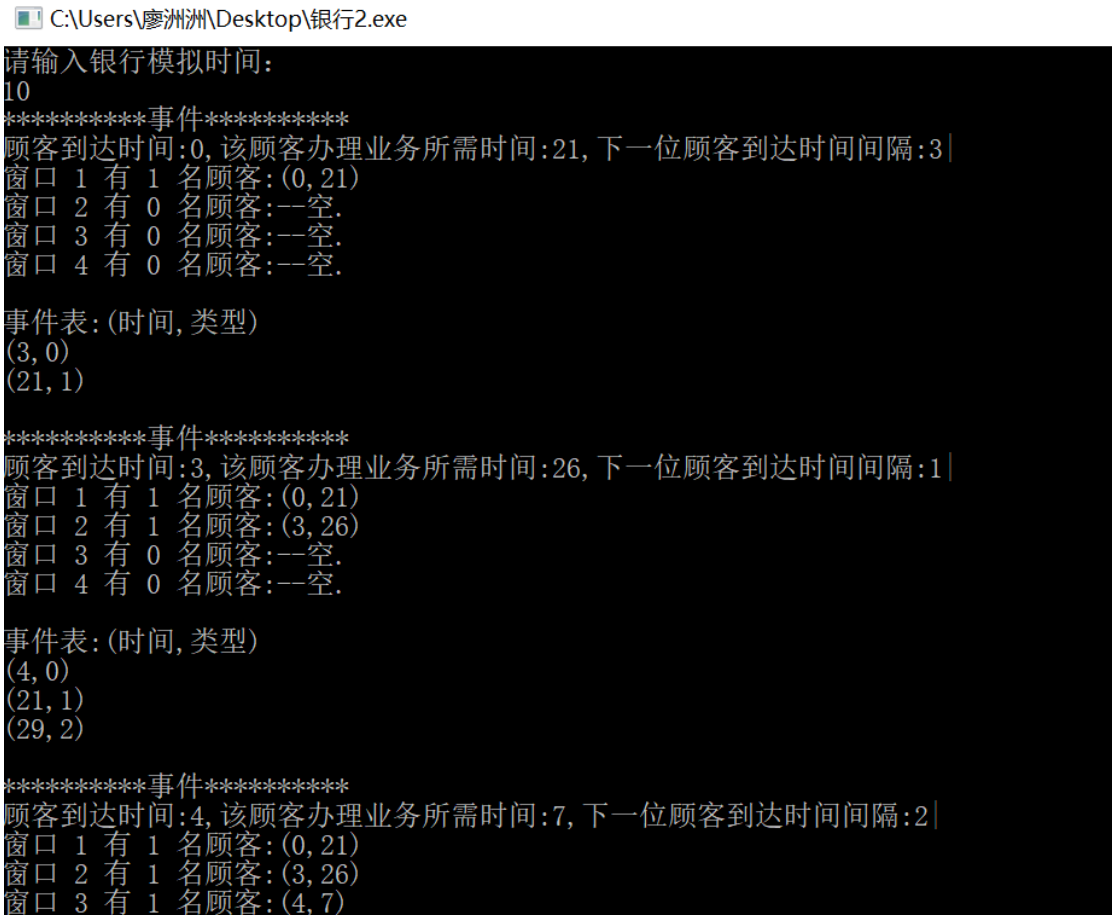
顾客离开，将其取出队列，同时得到该队列下一人的离开时间，将其的离开插入事件表

```
void CustomerDepature(){
    //顾客离开事件
    int i=en.NType;
    DelQueue(&q[i],&customer);
    printf("一位顾客离开，现在时间:%d\n",en.OccurTime);//输出顾客离开时间
    TotalTime+=en.OccurTime-customer.ArriveTime;
    if(!EmptyQueue(&q[i])){
        GetHead(&q[i],&customer);
        OrderInsert(&ev,NewEvent(en.OccurTime+customer.Duration,i));
    }
}
```

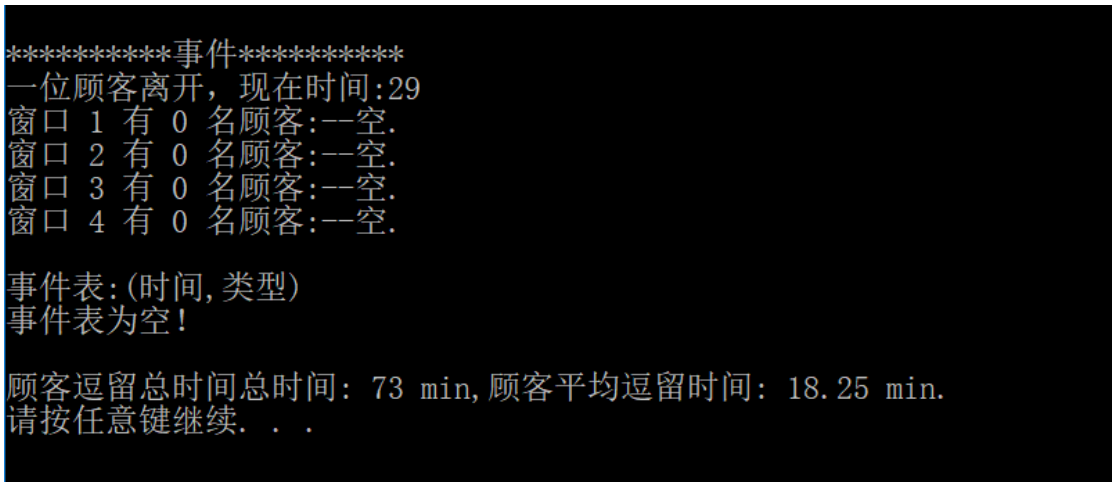
事件的插入，找到发生在其前面的最后一个事件，将其插入到事件表中

```
    *newptr=e;
    if(TRUE==ListEmpty(L)){                    //链表为空
        L->head->next=newptr;
        L->tail=newptr;
        newptr->next=NULL;
        return OK;
    }
    q=L->head;
    p=L->head->next;
    while(p){                                  //遍历整个链表
        if(p->OccurTime>=newptr->OccurTime)
            break;
        q=p;
        p=p->next;
    }
    q->next=newptr;
    newptr->next=p;
    if(!p)                                     //插入位置为链表尾部
        L->tail=newptr;
    return OK;
}
```

4. 实验结果及分析



输入银行模拟事件，程序会将所有事件的发生过程打印出来



最后程序会输出客户逗留的总时间和平均时间

5. 实验小结

- 通过本次实验我掌握了栈、队列的表示和实现技术

- 掌握了如何运用栈、队列的特点来构建算法
- 同时学会了离散事件的模拟方法