数据结构上机报告

题目：银行业务模拟

班级：04班 姓名：王式政 学号：19160100027 完成日期：2020.11.01

一、需求分析

1.问题描述

客户业务分2种，第一种是申请从银行得到一笔资金，即借款或取款。第2种是向银行投入一笔资金，即存款或还款。银行有2个服务窗口，相应地有2个队列。客户到达银行后先排第一个队。处理每个客户业务时，如果属于第一种，且申请额超出银行现存资金总额尔得不到要求，则立刻排入第2个队等候，直至满足时才离开银行，否则业务处理完后立刻离开银行。没接待完一个第二种业务的客户，则顺序检查和处理（如果可能）第二个队列中的得客户，对能满足的申请者予以满足，不能满足者重新排到第二个队列的队尾。注意，再此检查过程中，一旦银行资金总额少于或等于刚才第一个队列中最后一个客户（第2中业务）被接待之前的数额，或者本次以将第2个队列检查或者处理了一遍，就停止检查（因为此时已不可能还有满足者）转而继续接待第一个队列的客户。任何时刻都只能开一个窗口。假设检查不需要时间。营业时间结束时所有客户立即离开银行。

写一个上述银行业务的事件驱动模拟系统，通过模拟方法求出客户在银行内逗留的平均时间。

1. 基本要求

利用动态存储结构实现模拟。

3.测试数据

一天营业开始时银行拥有的款额为10000(元)，营业时间为600(分钟)。其他模拟参量自定，注意测定两种极端的情况:一是两个到达事件之间的间隔时间很短，而客户的交易时间很长，另一个恰好相反，设置两个到达事件的间隔时间很长，而客户的交易时间很短。

4.必要假设

可以假设客户入行间隔时间为30分钟，银行业务处理时间为30分钟。

二、概要设计:

程序用两个循环队列作为存储银行两个柜台客户处理情况的数据结构类型，利用队列的入队、出队、遍历等操作实现银行业务的模拟。

循环队列实现

1. 定义一个结构体类型存储客户信息，一个队列类型存储柜台客户的处理情况。

typedef struct customer {

int money;

int waittime = 0;

int reachtime = 0;

}c;

typedef struct queue {

c a[1000];

int front=0;

int rear=0;

int length = 0;

}q;

2、主程序流程

a.读入基本数据参量，然后将给定的客户数据数据读入到队列中。

b.在读入过程中对数据做进一步处理，计算银行余额及客户等待时间。

c.得到银行的最终余额和客户等待的总时间，然后计算出客户等待的平均时间并输出。

三、详细设计

循环队列实现伪码

1. 函数定义及功能

void queue2(q \*s,int time,int \*subtime,int \*m,int dealtime,int \*sumt) {

if(\*subtime < time && s->length > 0){

int i,t;

for(i = 0;i < s->length;i++) {

if((s->a[s->front].money + \*m) >= 0) {

\*subtime -= dealtime;

if(\*subtime < 0) return;

s->a[s->front].waittime += dealtime;

\*m += s->a[s->front].money;

t = out(s);

\*subtime += t;

\*sumt += t;

}

else {

in(s->a[s->front],s);

out(s);

}

}

}

if(\*subtime >= time && s->length > 0) {

int i,t;

for(i = s->front;i < s->rear;i++) s->a[i].waittime = s->a[s->front].reachtime - time;

\*subtime = time;

for(i = 0;i < s->length;i++) {

if((s->a[s->front].money + \*m) >= 0) {

\*subtime -= dealtime;

if(\*subtime < 0) return;

s->a[s->front].waittime += dealtime;

\*m += s->a[s->front].money;

t = out(s);

\*subtime += t;

\*sumt += t;

}

else{

in(s->a[s->front],s);

out(s);

}

}

}

}

功能：处理第二个队列的等待客户，并计算出各个客户的等待时间并记录到总时间中。

void queue1(q \*p,q \*s,int dealtime,int nexttime,int \*m,int time,int dealmoney,int \*sumt) {

int t = nexttime;

c n;

n.money = dealmoney;

n.waittime = 0;

in(n,p);

while(nexttime > 0 && p->length > 0) {

if((p->a[p->front].money + \*m) >= 0 && p->a[p->front].waittime + nexttime >= dealtime) {

\*m += p->a[p->front].money;

nexttime -= (dealtime - p->a[p->front].waittime);

if(p->a[p->front].money > 0) queue2(s,time+nexttime-t,subtime,m,dealtime,sumt);

out(p);

\*sumt += dealtime;

}

if((p->a[p->front].money + \*m) < 0) {

in(p->a[p->front],s);

s->a[s->front].reachtime = time+nexttime-t;

out(p);

}

else if(p->a[p->front].waittime + nexttime < dealtime) {

p->a[p->front].waittime += nexttime;

nexttime = 0;

}

}

}

功能：处理第一个队列的等待客户，计算出各个客户的等待时间并记录到总时间中，并将无法满足要求的客户置入第二个队列，并在银行余额增加后处理第二队列。

主程序处理部分：

1.读入数据与计算部分

printf("请输入银行持有钱数（元）：");

scanf("%d",&money);

printf("请输入营业时间（分钟）：");

scanf("%d",&time);

printf("请输入客户入行间隔时间（分钟）：");

scanf("%d",&nexttime);

printf("请输入银行业务处理时间（分钟）：");

scanf("%d",&dealtime);

subt = time;

while(time > 0) {

sumc++;

printf("请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：");

scanf("%d",&dealmoney);

queue1(s,p,dealtime,nexttime,m,time,dealmoney,sumt); /\*进行队列1的数据处理\*/

time -= nexttime;

}

while(p->length) sumtime+=out(p); //计算未处理用户的等待时间

while(s->length) sumtime+=out(s) + subt;

printf("银行所剩余额为：%d（元）\n客户平均滞留时间为：%.2f分钟）",\*m,\*sumt\*1.0 / sumc);

四、调试分析

1、调试问题

a.初期实现程序时，没有考虑到队列2中因空置引起的等待时间，后经过输入数据调试分析发现并解决了问题。

b.处理数据时额外计入了部分营业时间外的客户存取款数据，后增加了语句判断，解决了该问题。

2、算法的时间复杂度分析和改进设想

算法的时间复杂度为O（N^2），N为客户数量，目前存在程序可视性较差（没有图形界面，不够直观），时空复杂度较高等问题，后期导入图形处理模块、完善时间计算算法可以提高程序的可用性。

1. 经验和体会

对循环队列的出队入队操作，指针运算，以及动态存储结构的处理有了更深的了解。对于有固定的输入顺序，大小规模比较稳定的数据，使用顺序队列存储，既方便操作又容易调试修改。

五、用户使用说明

用户只需运行所给文件夹下exe程序，按照给定的顺序输入数据，即可得到输岀结果。

六、测试结果

输入样例

10000

600

30

30

5000

2000

-18000

-10000

600

800

-200

114514

-5000

800

-500

1000

20000

-8000

-6000

7000

200

-50000

2000

8000

输出样例

请输入银行持有钱数（元）：10000

请输入营业时间（分钟）：600

请输入客户入行间隔时间（分钟）：30

请输入银行业务处理时间（分钟）：30

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：5000

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：2000

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：-18000

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：-10000

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：600

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：800

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：-200

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：114514

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：-5000

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：800

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：-500

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：1000

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：20000

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：-8000

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：-6000

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：7000

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：200

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：-50000

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：2000

请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：8000

银行所剩余额为：74214（元）

客户平均滞留时间为：39.00分钟

1. 附录

#include<stdio.h>

int subt = 0;

int \*subtime = &subt;

typedef struct customer {

int money;

int waittime = 0;

int reachtime = 0;

}c;

typedef struct queue {

c a[1000];

int front=0;

int rear=0;

int length = 0;

}q;

int in(c n,q \*s) {

if(s->rear == 1000) s->rear = 0;

s->a[s->rear].money = n.money;

s->a[s->rear].waittime = n.waittime;

s->a[s->rear].reachtime = n.reachtime;

s->rear++;

s->length++;

}

int out(q \*s) {

int n=s->a[s->front].waittime;

s->front++;

if(s->front == 1000) s->rear = 0;

s->length--;

return n;

}

void queue2(q \*s,int time,int \*subtime,int \*m,int dealtime,int \*sumt) {

if(\*subtime < time && s->length > 0){

int i,t;

for(i = 0;i < s->length;i++) {

if((s->a[s->front].money + \*m) >= 0) {

\*subtime -= dealtime;

if(\*subtime < 0) return;

s->a[s->front].waittime += dealtime;

\*m += s->a[s->front].money;

t = out(s);

\*subtime += t;

\*sumt += t;

}

else {

in(s->a[s->front],s);

out(s);

}

}

}

if(\*subtime >= time && s->length > 0) {

int i,t;

for(i = s->front;i < s->rear;i++) s->a[i].waittime = s->a[s->front].reachtime - time;

\*subtime = time;

for(i = 0;i < s->length;i++) {

if((s->a[s->front].money + \*m) >= 0) {

\*subtime -= dealtime;

if(\*subtime < 0) return;

s->a[s->front].waittime += dealtime;

\*m += s->a[s->front].money;

t = out(s);

\*subtime += t;

\*sumt += t;

}

else{

in(s->a[s->front],s);

out(s);

}

}

}

}

void queue1(q \*p,q \*s,int dealtime,int nexttime,int \*m,int time,int dealmoney,int \*sumt) {

int t = nexttime;

c n;

n.money = dealmoney;

n.waittime = 0;

in(n,p);

while(nexttime > 0 && p->length > 0) {

if((p->a[p->front].money + \*m) >= 0 && p->a[p->front].waittime + nexttime >= dealtime) {

\*m += p->a[p->front].money;

nexttime -= (dealtime - p->a[p->front].waittime);

if(p->a[p->front].money > 0) queue2(s,time+nexttime-t,subtime,m,dealtime,sumt);

out(p);

\*sumt += dealtime;

}

if((p->a[p->front].money + \*m) < 0) {

in(p->a[p->front],s);

s->a[s->front].reachtime = time+nexttime-t;

out(p);

}

else if(p->a[p->front].waittime + nexttime < dealtime) {

p->a[p->front].waittime += nexttime;

nexttime = 0;

}

}

}

int main() {

int dealtime,nexttime,money,time,dealmoney;

int sumtime = 0,sumc = 0,\*sumt = &sumtime;

int \*m = &money;

q a,b,\*s = &a,\*p = &b;

printf("请输入银行持有钱数（元）：");

scanf("%d",&money);

printf("请输入营业时间（分钟）：");

scanf("%d",&time);

printf("请输入客户入行间隔时间（分钟）：");

scanf("%d",&nexttime);

printf("请输入银行业务处理时间（分钟）：");

scanf("%d",&dealtime);

subt = time;

while(time > 0) {

sumc++;

printf("请输入客户账户额度变化，存款与还钱为正，取款与借钱为负：");

scanf("%d",&dealmoney);

queue1(s,p,dealtime,nexttime,m,time,dealmoney,sumt);

time -= nexttime;

}

while(p->length) sumtime+=out(p);

while(s->length) sumtime+=out(s) + subt;

printf("银行所剩余额为：%d（元）\n客户平均滞留时间为：%.2f分钟）",\*m,\*sumt\*1.0 / sumc);

return 0;

}