实 验 报 告

学院： 计算机与信息学院 专业： 软件二班 年级： 大二

姓名： 吴生杰 学号： 3236016046 实验室号： 401 计算机号：91

实验日期： 2024 年 11 月26 日 指导教师签字： 成绩：

报告退发 (订正 、 重做)

实验三 排队问题的系统仿真

1. 实验目的：

1.掌握循环队列的存储结构；

2.掌握队列的初始化、入队、出队和判队空等基本运算算法；

3.学习利用队列实际电子秤阻尼算法的仿真；

1. 实验内容

**1.题目内容**：使用队列模拟理发馆的排队现象，通过仿真手法评估其营业状况。

**2.基本要求：**设某理发馆有N把理发椅，**可同时为N位顾客进行理发**。当顾客进门时，若有空椅，则可立即坐下理发，否则需依次排队等候。一旦有顾客理完离去时，排在队头的顾客便可开始理发。若理发馆每天连续营业T小时，求一天内顾客在理发馆内的平均逗留时间、顾客排队等候理发的队列长度平均值、营业时间到点后仍需完成服务的收尾工作时间。

3 需求分析

1. 本程序模拟理发馆排队现象。当给定理发椅数及营业时间后，由随机数确定顾客理发时间及进门间隔时间，可求出一天内顾客在理发馆平均逗留时间，平均队长及关门后收尾工作的时间。
2. 本程序由用户读入的数据仅为理发椅数及营业时间。营业的时间以分钟计，理发椅数及关门时间均为整型，且均大于等于1。
3. 运行程序后，得到结果为顾客数、平均等候时间、平均队长和收尾工作的时间。输出结果：

Number of customer: CustomerNum

Average time: Totaltime/CustomerNum

Average queuelength: Totallength/CustomerNum

Addition time: t-CloseTime

仿真伪代码：

Simulation{

设定事件表中的第一个元素；// 第一个入店的顾客

置空队；// 创建空的队列（无人排队）

while(事件不为空){ // 只要

从事件中删除发生时刻最早的元素； // 从理发店中删除最早理发完的人 **（排序）**

if(事件类型=0){ // **如果还没关门**

//处理顾客进门事件 🡪 产生两个随机数

累计顾客人数；// 进店来顾客数量就++

if(下一顾客到达时刻<关门时刻) 进门事件插入事件表；

**if(有空闲理发椅){**

**新出门事件插入事件表；**

**累计顾客逗留时间；**

**}**

**else {**

**当前顾客插入队尾；**

**累计队列长度；**

**}**

}

else {

事件类型=1，处理顾客离开事件 // **事件类型1就是已经到了关门的时刻了**

if(队不空){

删除队头元素；

记录顾客离开的最晚时间；

新的出门事件插入事件表；

累计顾客逗留时间；

}//if

}//else

}//while

**计算平均队长；**

**计算平均逗留时间；**

**计算收尾工作的时间；**

}//Sumulation

**4测试数据：**理发椅数目N及关门时间由用户读入，第一个顾客进门的时刻为0，之后每个顾客的进门时刻在前一个顾客进门时设定。即进门事件发生时随即产生两个随机数(durtime，intertime)，durtime为进门顾客理发所需要的服务时间(简称理发时间)；intertime为下一个顾客将到达的时间间隔(简称间隔时间)。R为由随机数发生器产生的随机数，顾客理发时间和顾客之间的间隔时间假设与R有关，由下式确定：

Durtime=15+R%50

Intertime=2+R%10

例：

输入数据：

Input the chairs’ number: 7

Input CloseTime:

运行结果输出：

Number of customer: 77

Average time: 43

Average queuelength: 0.87

Addition time:36

1. 实验环境

1.硬件：微型计算机、局域网；

2.软件：Windows操作系统、Dev-C++或VC++或C++Builder集成开发环境；

四、实验步骤

#include <iostream>

#include <queue>

#include <stdlib.h>

#include <ctime>

using namespace std;

struct *Customers*

{

  int cut\_time;

  int interval\_time; *// 下一个顾客到来的时间*

  int end\_time; *// 当前顾客结束理发的时刻*

  int intime; *// 入队时间*

};

struct *CompareEndTime*

{

  bool *operator()*(const *Customers* &*a*, const *Customers* &*b*)

  {

    return *a*.end\_time > *b*.end\_time;

*// end\_time 小的优先排在队首，因此用 >*

  }

};

*queue*<*Customers*> outside;

*priority\_queue*<*Customers*, *vector*<*Customers*>, *CompareEndTime*> inside;

int sum\_time, sum\_chair;

int *main*()

{

  cin *>>* sum\_chair *>>* sum\_time;

*srand*(*time*(0)); *// 用当前时间设置随机数种子*

  int durTime = 15 + *rand*() % 50;

  int interTime = 2 + *rand*() % 10;

*Customers* c = {durTime, interTime, durTime, 0};

  inside.*push*(c);

*// 赋值给第一个客人*

  int interval = c.interval\_time;

  int curtime = 0; *// 当前时间*

  int curchair = sum\_chair; *// 当前椅子剩余数量*

  curchair--; *// 第一个顾客进来*

  int sign = 0; *// 理发店未关门*

  int cnt = 1;

  int waitTimeSum = 0; *// 等候时长*

  int queue\_len = 0;

  while (!sign)

  {

    queue\_len += outside.*size*();

    while (inside.*size*() && inside.*top*().end\_time == curtime)

    {

*// 同时理完发(while)*

      inside.*pop*(); *// 弹出队头*

      curchair++;

    }

    if (curtime == interval) *// 如果理发店未关门 并且下一个顾客到来了*

    {

      int DurTime = 15 + *rand*() % 50;

      int InterTime = 2 + *rand*() % 10;

*Customers* C = {DurTime, InterTime, curtime + DurTime, curtime};

      outside.*push*(C);

      cnt++;

      interval = C.interval\_time + curtime; *// 当前的顾客的下一个顾客到来的时刻*

    }

    while (!sign && curchair && outside.*size*()) *// 如果当前椅子有空闲 且 外面有队伍*

    {

      outside.*front*().end\_time = curtime + outside.*front*().cut\_time; *// 更新当前队头的结束时间*

      inside.*push*(outside.*front*());

      waitTimeSum += curtime - outside.*front*().intime + outside.*front*().cut\_time;

      outside.*pop*();

      curchair--;

    }

    curtime++; *// 时间流逝*

    if (curtime == sum\_time)

    {

      sign = 1; *// 理发店不接受顾客进门*

      break;

    }

  }

*//   int n = 0;*

*//   while(inside.size())*

*//   {*

*//       cout<<inside.top().end\_time<<" ";*

*//       inside.pop();*

*//   }*

*//   cout<<endl;*

*//   while(outside.size())*

*//   {*

*//       cout<<outside.front().end\_time<<" ";*

*//       outside.pop();*

*//   }*

  while (inside.*size*() || outside.*size*()) *// 如果理发店关门 and 还有顾客*

  {

    queue\_len += outside.*size*();

    while (inside.*size*() && inside.*top*().end\_time == curtime)

    {

*// 同时理完发(while)*

      inside.*pop*(); *// 弹出队头*

      curchair++;

    }

    while (curchair && outside.*size*()) *// 将当前还在排队的人弹入理发队列*

    {

      outside.*front*().end\_time = curtime + outside.*front*().cut\_time; *// 更新当前队头的结束时间*

      inside.*push*(outside.*front*());

      waitTimeSum += curtime - outside.*front*().intime + outside.*front*().cut\_time;

      outside.*pop*();

      curchair--;

    }

    curtime++;

  }

*// ans*

  cout *<<* cnt *<<* *endl*; *// 顾客数量*

  cout *<<* waitTimeSum / cnt *<<* *endl*; *// 平均等候时长*

  cout *<<* 1.0 \* queue\_len / curtime *<<* *endl*; *// 平均队长（每一个时刻的队伍长度总和/总时刻）;*

  cout *<<* curtime - sum\_time - 1 *<<* *endl*; *// 收尾工作时间*

*// @test*

*//  cout<<sum\_time<<' '<<curtime;*

  return 0;

}

五、实验调试与结果分析

（描述实验的调试过程，实验中发生的现象、中间结果、最终得到的结果，并进行分析说明，分析可能的误差或错误原因等）

六、总结

（说明实验过程中遇到的问题及解决办法；新发现或个人的收获；未解决/需进一步研讨的问题或建议新实验方法等）