|  |  |
| --- | --- |
| **Author Identification Block** | |
| **Author:** | Chris Graff |
| **Student ID:** | \*20274911 |
| **E-Mail:** | [cgraff@uco.edu](mailto:cgraff@uco.edu) |
| **Course:** | CMSC 2613 – Programming 2 |
| **CRN:** | 21641, Spring 2012 |
| **Project:** | p05 |
| **Due:** | February 24, 2012 |
| **Account:** | tt025 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Scoring Block** | | | |
| **Component** | **Available** | **Earned** | **Explanation** |
| Compilation |  |  |  |
| Submission Instructions | 2 |  |  |
| Author Identification | 1 |  |  |
| Modularity | 3 |  |  |
| Command Line | 3 |  |  |
| Input file | 3 |  |  |
| Output file | 3 |  |  |
| Execution | 10 |  |  |
| **Total** | **25** |  |  |

#----------------------------------------------------------

#File p05make that creates executable file p05

#----------------------------------------------------------

#Author: Thomas R. Turner

#E-Mail: trturner@ucok.edu

#Date: September, 2005

#----------------------------------------------------------

#Copyright September, 2005 by Thomas R. Turner

#Do not reproduce without permission from Thomas R. Turner

#----------------------------------------------------------

# Define executable files

#----------------------------------------------------------

obj = p05.o \

Sim05.o \

Event05.o \

ExponentialElapsed05.o \

Exponential05.o \

Uniform05.o

#----------------------------------------------------------

#Create executable file p05

#----------------------------------------------------------

p05: ${obj}

g++ -o p05 ${obj} -lm

#----------------------------------------------------------

#Create p05.o that exercises class Event

#----------------------------------------------------------

p05.o: p05.cpp Sim05.h

g++ -c -g p05.cpp

#----------------------------------------------------------

#Create the simulation

#----------------------------------------------------------

Sim05.o: Sim05.cpp Sim05.h Queue05.h Event05.h

g++ -c -g Sim05.cpp

#----------------------------------------------------------

#Create Event05.o that implements member function of

#class Event

#----------------------------------------------------------

Event05.o: Event05.cpp \

Event05.h \

Queue05.h \

ExponentialElapsed05.h

g++ -c -g Event05.cpp

#----------------------------------------------------------

#Create ExponentialElapsed05.o that implements member

#functions of class ExponentialElapsed

#----------------------------------------------------------

ExponentialElapsed05.o: ExponentialElapsed05.cpp \

ExponentialElapsed05.h \

Exponential05.h

g++ -c -g ExponentialElapsed05.cpp

#----------------------------------------------------------

#Create Exponential05.o that implements member functions

#of class Exponential

#----------------------------------------------------------

Exponential05.o: Exponential05.h \

Uniform05.h

g++ -c -g Exponential05.cpp

#----------------------------------------------------------

#Create Uniform05.o that implements the uniform distribution

#----------------------------------------------------------

Uniform05.o: Uniform05.cpp Uniform05.h

g++ -c -g Uniform05.cpp

//----------------------------------------------------------------

//Author: Chris Graff

//StudentID#: \*20274911

//Email: cgraff@uco.edu

//Course: CMSC2613 Programming II

//CRN: 21641, Spring 2012

//Project: p05

//Due: Februaru 24th, 2012

//Account: tt025

//----------------------------------------------------------------

#include<iostream>

#include<string.h>

#include<iomanip>

#include<fstream>

#include "Event05.h"

#include "Uniform05.h"

#include "Exponential05.h"

#include "ExponentialElapsed05.h"

#include "Queue05.h"

#include "Sim05.h"

using namespace std;

struct CommandLineException

{

CommandLineException (int max, int actual)

{

cout <<endl <<"Too many command line arguements." <<endl;

cout <<"A maximum of " <<max <<" arguements are permitted." <<endl;

cout <<actual <<" arguements were entered." <<endl;

}

};

struct FileException

{

FileException (char\* filename)

{

cout <<endl <<"File " <<filename <<" could not be opened or doesn't exist" <<endl;

}

};

int main (int argc, char\* argv[])

{

try

{

char iFileName[255], oFileName[255];

switch (argc)

{

case 1:

cout <<"Enter the input file name:";

cin >> iFileName;

cout <<"Enter the output file name:";

cin >> oFileName;

break;

case 2:

strcpy(iFileName, argv[1]);

cout <<"Enter the output file name:";

cin >> oFileName;

break;

case 3:

strcpy(iFileName, argv[1]);

strcpy(oFileName, argv[2]);

break;

default:

throw CommandLineException (2, argc-1);

break;

}

ifstream i(iFileName);

if (!i)

throw FileException(iFileName);

ofstream o(oFileName);

if (!o)

throw FileException(oFileName);

Sim S;

S.Run(i, o);

i.close();

o.close();

}

catch (...)

{

cout <<"Program terminated." <<endl;

}

}

#ifndef Sim05\_h

#define Sim05\_h

//----------------------------------------------------------------

//Author: Chris Graff

//StudentID#: \*20274911

//Email: cgraff@uco.edu

//Course: CMSC2613 Programming II

//CRN: 21641, Spring 2012

//Project: p05

//Due: February 24th, 2012

//Account: tt025

//----------------------------------------------------------------

#include "Queue05.h"

#include <string>

#include <fstream>

#include <iostream>

using namespace std;

class Sim

{

Queue<double> Q;

double maxl; //longest line length

double totpro; //sum of products of line length

double maxres; //longest wait

double total; //total wait time

int served; //number served

double arint; //mean time between arrivals

double arate; //arrival rate

double depint; //mean time between departures

double derate; //departue rate

double extime; //expected response

double repvar; //response variance

double avrep; //average response time

double exlen; //expected line length

double lenvar; //line length variance

double avlen; //average length of line

int simlim; //time limit

double rho; //traffic intensity

double duration; //duration of simulation

void arrival(int t);

void depart(int t);

void printitem(string first, double second, int third, ostream& o);

void printitem(string first, int second, ostream& o);

void printall(ostream& o);

public:

Sim();

void Run(istream& i, ostream& o);

};

#endif

//----------------------------------------------------------------

//Author: Chris Graff

//StudentID#: \*20274911

//Email: cgraff@uco.edu

//Course: CMSC2613 Programming II

//CRN: 21641, Spring 2012

//Project: p05

//Due: February 24th, 2012

//Account: tt025

//----------------------------------------------------------------

#include "Queue05.h"

#include <fstream>

#include <cstdlib>

#include "Sim05.h"

#include <iomanip>

#include <math.h>

#include "Event05.h"

#include "Uniform05.h"

#include "Exponential05.h"

#include "ExponentialElapsed05.h"

using namespace std;

struct QExcept

{

QExcept(string m)

{

cout << endl << "The Queue is " << m << "." << endl;

}

};

struct RHOprob

{

RHOprob(double rho)

{

cout << endl;

cout << "Crap, rho is " << rho << " and i dont know why.";

cout << endl;

}

};

Sim::Sim()

{

maxl = maxres = total = arint = totpro = arate = depint = 0.0;

derate = extime = repvar = avrep = exlen = lenvar = 0.0;

avlen = rho = duration = 0.0;

simlim = served = 0;

}

void Sim::arrival(int t)

{

if(Q.full()) return;

Q.enq(t);

if(Q.getlength() > maxl) maxl = Q.getlength();

}

void Sim::depart(int t)

{

if(Q.empty()) return;

int d = t - Q.deq();

if (d > maxres) maxres = d;

total += d;

++served;

}

void Sim::printitem(string first, double second, int third, ostream& o)

{

o << setw(40) << left << first;

o << fixed;

o << setw(10) << setprecision(2) << right << second;

string thing;

switch(third)

{

case 0:

thing = "minutes";

break;

case 1:

thing = "customers";

break;

case 2:

thing = "customers/hour";

break;

default:

thing = "messed this stuff up....again";

break;

}

o << setw(20) << right << thing << endl;

}

void Sim::printitem(string first, int second, ostream& o)

{

o << setw(40) << left << first;

o << setw(10) << right << second;

o << setw(20) << right << "customers" << endl;

}

void Sim::printall(ostream& o)

{

printitem("Mean time between arrivals", arint, 0, o);

printitem("Arrival rate", 60\*arate, 2, o);

o << endl;

printitem("Mean time between departures", depint, 0, o);

printitem("Departure rate", 60\*derate, 2, o);

o << endl;

printitem("Expected response time", extime, 0, o);

printitem("Response time variance", repvar, 0, o);

printitem("Simulation response time", avrep, 0, o);

o << endl;

printitem("Expected line length", exlen, 1, o);

printitem("Line length variance", lenvar, 1, o);

printitem("Simulation mean line length", avlen, 1, o);

o << endl;

printitem("Customers served", served, o);

printitem("Simulation maximum response time", maxres, 0, o);

printitem("Simulation maximum line length", maxl, o);

o << endl;

printitem("Simulation lasted for", duration, 0, o);

}

void Sim::Run(istream& i, ostream& o)

{

i >> arint >> depint >> simlim;

arate = (1 / arint);

derate = (1 / depint);

rho = arate / derate;

if (rho > 1) throw RHOprob(rho);

exlen = rho / (1-rho);

extime = exlen / arate; //problem

lenvar = rho / pow((1-rho), 2);

repvar = lenvar / arate; //problem

Event event (arate, derate);

duration = 0.0;

for (;;)

{

event.Occur();

if(event.Time() > simlim || duration > simlim)

break;

double interval = event.Time() - duration;

int num = Q.getlength();

totpro += num \* interval;

duration = event.Time();

if (event.IsArrival())

arrival(event.Time());

else

depart(event.Time());

}

avrep = (double) total / served; //problem

avlen = totpro / duration;

printall(o);

}

#ifndef Queue05\_h

#define Queue05\_h

//----------------------------------------------------------------

//Author: Chris Graff

//StudentID#: \*20274911

//Email: cgraff@uco.edu

//Course: CMSC2613 Programming II

//CRN: 21641, Spring 2012

//Project: p05

//Due: February 24th, 2012

//Account: tt025

//----------------------------------------------------------------

#include <cstdlib>

#include <string>

#include <iomanip>

#include <iostream>

using namespace std;

template <typename T>

class Queue

{

struct Element

{

Element\* newer;

T value;

Element(T v):newer(0), value(v){}

};

struct QExcept

{

QExcept(string m)

{

cout << endl;

cout << "I am the Queue and I am " << m;

cout << "." << endl;

}

};

Element\* oldest;

int count;

Element\* newest;

void Kill(Element\* e)

{

while(e)

{

Element\* p = e;

e = e->newer;

delete p;

}

}

public:

Queue():oldest(0), count(0), newest(0){}

~Queue(){Kill(oldest);}

bool full(){return false;}

T getlength(){return count;}

bool empty(){return count <= 0;}

void enq(T v)

{

if(full())

{

throw QExcept("full");

return;

}

Element\* e = new Element(v);

if(empty())

oldest = e;

else

newest->newer = e;

newest = e;

++count;

}

T deq()

{

if(empty())

{

throw QExcept("empty");

return 0;

}

Element\* e = oldest;

T v = e -> value;

oldest = oldest -> newer;

--count;

if(empty())

newest = 0;

delete e;

return v;

}

};

#endif

#ifndef Uniform05\_h

#define Uniform05\_h 1

//--------------------------------------------------------------------------

//File Uniform05.h defines class Uniform used to obtain random samples from

//a uniform distribution

//--------------------------------------------------------------------------

//Author: Thomas R. Turner

//E-Mail: trturner@ucok.edu

//Date: October, 2003

//--------------------------------------------------------------------------

//Copyright October, 2003 by Thomas R. Turner

//Do not reproduce without permission from Thomas R. Turner

//--------------------------------------------------------------------------

//Standard C and C++ include files

//--------------------------------------------------------------------------

class Uniform {

public:

Uniform(); //Constructor used to seed the uniform distribution

double Sample(void); //Obtain a sample from the uniform distribution

};

#endif

//------------------------------------------------------------------------

//File Uniform05.cpp implements member functions of class Uniform

//------------------------------------------------------------------------

//Author: Thomas R. Turner

//E-Mail: trturner@ucok.edu

//Date: 2002

//--------------------------------------------------------------------------

//Copyright 2002 by Thomas R. Turner

//Do not reproduce without permission from Thomas R. Turner

//--------------------------------------------------------------------------

//Standard C and C++ include files

//------------------------------------------------------------------------

#include <ctime>

#include <cstdlib>

//------------------------------------------------------------------------

//Namespaces

//------------------------------------------------------------------------

using namespace std;

//------------------------------------------------------------------------

//Application include files

//------------------------------------------------------------------------

#include "Uniform05.h"

//------------------------------------------------------------------------

//Constructor

//Seed the random number generator

//------------------------------------------------------------------------

Uniform::Uniform()

{ time\_t t;

srand((unsigned)time(&t));

}

//------------------------------------------------------------------------

//Function Sample extracts a sample from the uniform distribution

//------------------------------------------------------------------------

double Uniform::Sample(void)

{ return (double)rand()/(double)RAND\_MAX;

}

#ifndef Event05\_h

#define Event05\_h 1

//-----------------------------------------------------------------------

//File Event05.h defines class Event.

//-----------------------------------------------------------------------

//Author: Thomas R. Turner

//E-Mail: trturner@ucok.edu

//Date: October, 2003

//-----------------------------------------------------------------------

//Copyright October, 2003 by Thomas R. Turner

//Do not reproduce without permission from Thomas R. Turner

//-----------------------------------------------------------------------

//C and C++ standard include files

//-----------------------------------------------------------------------

#include <iostream>

#include <string>

#include <fstream>

#include <iomanip>

//-----------------------------------------------------------------------

//Namespaces

//-----------------------------------------------------------------------

using namespace std;

//-----------------------------------------------------------------------

//Application include files

//-----------------------------------------------------------------------

#include "ExponentialElapsed05.h"

//-----------------------------------------------------------------------

//Type EventKind enumerates the kinds of events that are produced by

//class Event.

//-----------------------------------------------------------------------

enum EventKind{ek\_arrival,ek\_departure};

//-----------------------------------------------------------------------

//Class Event merges two streams of timed events. Arrivals are merged

//departures in strict time order.

//-----------------------------------------------------------------------

class Event {

enum State{initial,arrival,departure};

//-------------------------------------------------------------------

//Event Data consists of the kind of event, either an arrival or a

//departure, and the Time of the event.

//-------------------------------------------------------------------

struct Data {

EventKind EK; //Kind of event either an arrival or a

//departure

double Time; //Elapsed time since time=0 for the event.

//----------------------------------------------------------------

//Constructor initialized the initial event as an arrival at time

//zero.

//----------------------------------------------------------------

Data(EventKind ek=ek\_arrival,double t=0.0):EK(ek),Time(t){}

};

State S; //State of the last event.

Data E; //Data describing the current event

Data A; //Data describing the most recent

//arrival

Data D; //Data describing the most recent

//departure

ExponentialElapsed Arrival; //Time of the most recent arrival

ExponentialElapsed Departure; //Time of the most recent departure

//and departure times

public:

Event(double A,double D); //Initialize exponential distributions

//for arrivals and departures

void Occur(void); //Obtain the next arrival or departure

double Time(void); //Return the time of the current event

bool IsArrival(void); //Is the current event an arrival?

bool IsDeparture(void); //Is the current event a departure?

void Print(ostream& o); //Print an event.

};

#endif

//-----------------------------------------------------------------------

//File Event05.cpp implements class Event.

//-----------------------------------------------------------------------

//Author: Thomas R. Turner

//E-Mail: trturner@ucok.edu

//Date: October, 2003

//-----------------------------------------------------------------------

//Copyright October, 2003 by Thomas R. Turner

//Do not reproduce without permission from Thomas R. Turner

//-----------------------------------------------------------------------

//C and C++ standard include files

//-----------------------------------------------------------------------

#include <iostream>

#include <string>

#include <fstream>

#include <iomanip>

using namespace std;

//-----------------------------------------------------------------------

//Application include files

//-----------------------------------------------------------------------

#include "ExponentialElapsed05.h"

#include "Event05.h"

//-----------------------------------------------------------------------

//Constructor Event assigns the arrival rate a and the departure rate d

//to exponential distributions Arrival and Departure.

//-----------------------------------------------------------------------

Event::Event(double a,double d)

:S(initial)

,D()

,E()

,A()

,Arrival(a)

,Departure(d)

{}

//-----------------------------------------------------------------------

//Function Occur makes the next Event available. Events are queued in the

//order of occurrence.

//-----------------------------------------------------------------------

void Event::Occur(void)

{ switch (S) {

case initial:

A.Time=Arrival.Time();

A.EK=ek\_arrival;

D.Time=Departure.Time();

D.EK=ek\_departure;

break;

case arrival:

A.Time=Arrival.Time();

A.EK=ek\_arrival;

break;

case departure:

D.Time=Departure.Time();

D.EK=ek\_departure;

break;

}

if (A.Time<D.Time) {

E=A; S=arrival;

}else {

E=D;

S=departure;

}

}

//-----------------------------------------------------------------------

//Function Time returns the time of the current event.

//-----------------------------------------------------------------------

double Event::Time(void){return E.Time;}

//-----------------------------------------------------------------------

//Function IsArrival determines if the event is an arrival.

//-----------------------------------------------------------------------

bool Event::IsArrival(void){return E.EK==ek\_arrival;}

//-----------------------------------------------------------------------

//Function IsDeparture determines if the event is a departure.

//-----------------------------------------------------------------------

bool Event::IsDeparture(void){return E.EK==ek\_departure;}

//-----------------------------------------------------------------------

//Function Prints an event.

//-----------------------------------------------------------------------

void Event::Print(ostream& o)

{ o << endl;

o << setw(15) << fixed << setprecision(4) << Time();

o << " ";

if (IsArrival()) o << "arrival"; else o << "departurne";

}

#ifndef Exponential05\_h

#define Exponential05\_h 1

//-----------------------------------------------------------------------

//File Exponential05.h defines class Exponential.

//-----------------------------------------------------------------------

//Author: Thomas R. Turner

//E-Mail: trturner@ucok.edu

//Date: October, 2003

//-----------------------------------------------------------------------

//Copyright October, 2003 by Thomas R. Turner

//Do not reproduce without permission from Thomas R. Turner

//-----------------------------------------------------------------------

//C and C++ standard include files

//-----------------------------------------------------------------------

#include <iostream>

#include <string>

#include <fstream>

#include <iomanip>

#include <cmath>

//-----------------------------------------------------------------------

//Namespaces

//-----------------------------------------------------------------------

using namespace std;

//-----------------------------------------------------------------------

//Application include files

//-----------------------------------------------------------------------

#include "Uniform05.h"

//-----------------------------------------------------------------------

//class Exponential is designed to draw samples from the exponential CDF.

//-----------------------------------------------------------------------

class Exponential: public Uniform {

double L; //Lambda

public:

Exponential(double l); //Constructor

double Sample(void); //Sample

};

#endif

//-----------------------------------------------------------------------

//File Exponential05.cpp implements class Exponential.

//-----------------------------------------------------------------------

//Author: Thomas R. Turner

//E-Mail: trturner@ucok.edu

//Date: October, 2003

//-----------------------------------------------------------------------

//Copyright October, 2003 by Thomas R. Turner

//Do not reproduce without permission from Thomas R. Turner

//-----------------------------------------------------------------------

//C and C++ standard include files

//-----------------------------------------------------------------------

#include <iostream>

#include <string>

#include <fstream>

#include <iomanip>

#include <cmath>

//-----------------------------------------------------------------------

using namespace std;

//-----------------------------------------------------------------------

//Application include files

//-----------------------------------------------------------------------

#include "Uniform05.h"

#include "Exponential05.h"

//-----------------------------------------------------------------------

//Constructor

//-----------------------------------------------------------------------

Exponential::Exponential(double l):L(l){}

//-----------------------------------------------------------------------

//Function Sample returns a random sample from the exponential CDF

//-----------------------------------------------------------------------

double Exponential::Sample(void)

{ double s=-log(1.0-Uniform::Sample())/L;

return s;

}

#ifndef ExponentialElapsed05\_h

#define ExponentialElapsed05\_h 1

//-----------------------------------------------------------------------

//File ExponentialElapsed05.h defines class ExponentialElapsed.

//-----------------------------------------------------------------------

//Author: Thomas R. Turner

//E-Mail: trturner@ucok.edu

//Date: October, 2003

//-----------------------------------------------------------------------

//Copyright October, 2003 by Thomas R. Turner

//Do not reproduce without permission from Thomas R. Turner

//-----------------------------------------------------------------------

//C and C++ standard include files

//-----------------------------------------------------------------------

#include <iostream>

#include <string>

#include <fstream>

#include <iomanip>

//-----------------------------------------------------------------------

//Namespaces

//-----------------------------------------------------------------------

using namespace std;

//-----------------------------------------------------------------------

//Application include files

//-----------------------------------------------------------------------

#include "Exponential05.h"

//-----------------------------------------------------------------------

//class ExponentialElapsed produces an increasing sequence of values

//whose intervals are drawn from the Exponential distribution.

//-----------------------------------------------------------------------

class ExponentialElapsed: public Exponential {

double time; //Total elapsed time

public:

ExponentialElapsed(double l); //Constructor

double Time(void); //ExponentialElapsed time since time=0

};

#endif

//-----------------------------------------------------------------------

//File ExponentialElapsed05.cpp implements class ExponentialElapsed.

//-----------------------------------------------------------------------

//Author: Thomas R. Turner

//E-Mail: trturner@ucok.edu

//Date: October, 2003

//-----------------------------------------------------------------------

//Copyright October, 2003 by Thomas R. Turner

//Do not reproduce without permission from Thomas R. Turner

//-----------------------------------------------------------------------

//C and C++ standard include files

//-----------------------------------------------------------------------

#include <iostream>

#include <string>

#include <fstream>

#include <iomanip>

//-----------------------------------------------------------------------

//Namespaces

//-----------------------------------------------------------------------

using namespace std;

//-----------------------------------------------------------------------

//Application include files

//-----------------------------------------------------------------------

#include "Exponential05.h"

#include "ExponentialElapsed05.h"

//-----------------------------------------------------------------------

//Constructor ExponentialElapsed intializes member time to zero and

//establishes parameter L (lambda) in the exponential distribution

//F(t) = 1-exp(-Lt)

//-----------------------------------------------------------------------

ExponentialElapsed::ExponentialElapsed(double l):Exponential(l),time(0.0){}

//-----------------------------------------------------------------------

//Function Time returns the elapsed time since time zero for this sample

//from the exponential distribution.

//-----------------------------------------------------------------------

double ExponentialElapsed::Time(void)

{ double t=Exponential::Sample();

time+=t;

return time;

}