Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Моделирование

ОТЧЕТ

по лабораторной работе № 8

на тему

МЕТОД МОНТЕ\_КАРЛО

ВАРИАНТ № 1, 1, 1

Студент: П.В. Сякачёв

Проверила: Ю.О. Герман

МИНСК 2022

# 1. Цель работы

Научиться имитировать случайные события методом Монте-Карло.

# 2. Задание

## 2.1 Имитация случайных событий

* разработать алгоритм имитации для решения задачи на основе метода Монте-Карло.
* выполнить два испытания алгоритма, используя случайные числа из учебного пособия.
* разработать имитационную модель для решения задачи.

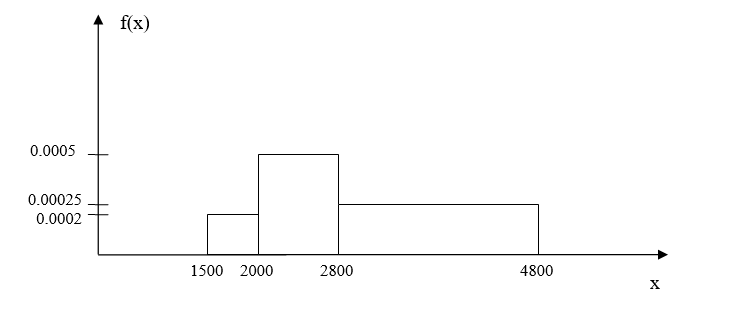
Предприятие выпускает некоторые изделия. Себестоимость изделия – 10 ден.ед. Изделия продаются по цене 15 ден.ед.

Изделие может иметь дефекты трех типов: A, B и C. Дефекты не связаны друг с другом, т.е. они возможны в любых комбинациях. Дефект A имеется у 6% изделий, дефект B – у 4%, дефект C – у 9%. Если изделие имеет один дефект, то он устраняется. Затраты на устранение дефекта A, B и C составляют, соответственно, 4, 6 и 2 ден.ед. Если изделие имеет два и более дефектов, то оно бракуется (т.е. в этом случае убытки от его выпуска составляют 10 ден.ед.).

Найти: а) среднюю прибыль от одного изделия; б) вероятность того, что изделие не будет забраковано.

## 2.2 Имитация случайных величин: метод обратных функций, метод исключений

Некоторая непрерывная случайная величина *X* задана плотностью распределения.



Требуется:

1) решить задачу имитации заданной случайной величины методом обратных функций. Для этого:

* разработать алгоритм имитации заданной случайной величины на основе метода обратных функций;
* используя случайные числа из учебного пособия (приложение 1), выполнить имитацию двух значений заданной случайной величины;
* выполнить проверку разработанного алгоритма, используя построение гистограммы;

2) решить задачу имитации заданной случайной величины методом исключений, выполнив те же действия, что и в пункте 1);

3) используя один из разработанных алгоритмов, выполнить имитацию заданной случайной величины и найти для нее: а) среднее значение; б) вероятность того, что величина *X* превысит некоторое заданное значение (выбрать это значение самостоятельно).

## 2.3 Решение задач моделирования объектов на основе имитации случайных величин

* разработать алгоритм имитации для решения задач на основе метода Монте-Карло.
* выполнить одно испытание алгоритма, используя случайные числа из учебного пособия (приложение 1).
* разработать имитационную модель на любом языке для решения задачи.

## 2.3.1 Часть 1

Предприятие выпускает комплекты инструментов по заказам. Количество инструментов в комплекте – от 3 до 6. Инструменты могут быть стандартными или высокоточными; комплект может содержать стандартные и высокоточные инструменты в любой комбинации. Время изготовления одного стандартного инструмента составляет от 30 мин до одного часа, высокоточного – от 40 мин до двух часов. Время испытания инструмента – экспоненциальная случайная величина со средним значением 10 мин для стандартных инструментов, 15 мин – для высокоточных.

Найти: а) среднее время изготовления и испытания комплекта; б) вероятность того, что комплект будет состоять только из высокоточных инструментов; в) вероятность того, что время изготовления и испытания комплекта превысит 24 часа.

## 2.3.2 Часть 2

В мастерской выполняется проверка и ремонт некоторых приборов. Прибор состоит из пяти схем. Вероятность неисправности схемы - 0,1. Работа с прибором включает следующие операции:

* проверка: от 2 до 5 мин на каждую схему;
* замена неисправных схем: время замены одной схемы – гауссовская случайная величина со средним значением 5 мин и стандартным отклонением 0,5 мин;
* настройка. Время настройки – экспоненциальная случайная величина со средним значением 6 мин, если ни одна схема не заменялась, и 10 мин – если потребовалась замена хотя бы одной схемы.

Найти: а) среднее время работы с прибором; б) вероятность того, что в приборе потребуется замена хотя бы одной схемы; в) среднее количество схем, заменяемых в одном приборе.

# 3. Ход работы

## 3.1 Общая часть выполнения

Для решения постановленных задач был выбран язык C#. Для получения наборов случайных чисел был написан класс Random\_functions с следующими методами:

* Get\_System\_Random() – получение случайных чисел, используя средства языка;
* Get\_Lehmer\_Random() – получение случайных чисел при помощи метода Лемера;
* Get\_Congruent\_Random() – получение случайных чисел при помощи конгруэнтного метода;
* Get\_Prepared1() – получение заранее заготовленных чисел из приложения 1;
* Get\_Prepared2() – получение заранее заготовленных чисел из приложения 1;
* Get\_Prepared3() – получение заранее заготовленных чисел из приложения 1;

## 3.2 Задание 1

Для выполнения задания 1 был написан класс Part\_1, вот основной его метод:

public void Run(double[] chances\_A, double[] chances\_B, double[] chances\_C)

{

int cost = 10;

int price = 25;

int cost\_A = 4, cost\_B = 6, cost\_C = 2;

double chance\_A = 0.06, chance\_B = 0.04, chance\_C = 0.09;

for (int i = 0; i < 100; i++)

{

money -= cost;

bool is\_A = false;

bool is\_B = false;

bool is\_C = false;

if (chances\_A[i] <= chance\_A) is\_A = true;

if (chances\_B[i] <= chance\_B) is\_B = true;

if (chances\_C[i] <= chance\_C) is\_C = true;

if (is\_A || is\_B || is\_C)

if ((!(is\_A && is\_B && is\_C)) && (is\_A ^ is\_B ^ is\_C))

{

if (is\_A) money -= cost\_A;

else if (is\_B) money -= cost\_B;

else if (is\_C) money -= cost\_C;

}

else

{

rejected++;

continue;

}

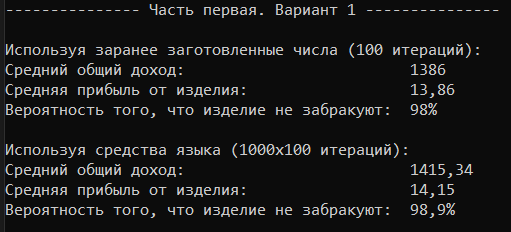
money += price;

}

count++;

}

Результат выполнения этой части программы:



## 3.3 Задание 2

Запишем уравнения плотности распределения:

Найдем выражения для функции распределения:

На интервале :

На интервале :

На интервале :

На интервале :

На интервале :

Таким образом функция распределения времени внесения добавок имеет вид:

Находим выражение :

Для выполнения задания 2 был написан класс Part\_2, вот два основных его метода:

public void Run\_1(double[] R)

{

double temp;

for (int i = 0; i < 100; i++)

{

if (R[i] < 0.1) temp = (R[i] + 0.3) / 0.0002;

else if (R[i] < 0.5) temp = (R[i] + 0.9) / 0.0005;

else temp = (R[i] + 0.2) / 0.00025;

X += temp;

if (temp > value) exceeding++;

count++;

}

}

public void Run\_2(double[] R1, double[] R2)

{

int a = 1500;

int b = 4800;

double fmax = 0.0005;

double Rs1, Rs2, FR;

for (int i = 0; i < 100; i++)

{

Rs1 = a + (b - a) \* R1[i];

Rs2 = fmax \* R2[i];

if (Rs2 >= 1500 && Rs2 <= 2000) FR = 0.0002;

else if (Rs2 > 2000 && Rs2 <= 2800) FR = 0.0005;

else FR = 0.00025;

if (Rs2 <= FR)

X += Rs1;

else continue;

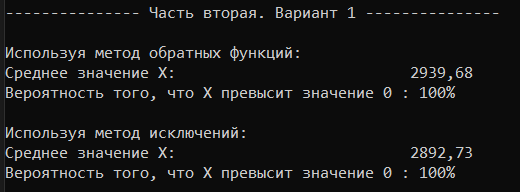
if (Rs1 > value) exceeding++;

count++;

}

}

Результат выполнения этой части программы:



Гистограмма значений X для метода обратных функций:

## 3.4 Задание 3

### 3.4.1 Часть 1

Для выполнения первой части третьего задания был написан класс Part\_3\_1, вот основной его метод:

public void Run(double[] Rand)

{

int t = 0;

for (int i = 0; i < 5; i++)

{

double cur\_time = 0;

bool is\_precise = true;

int number;

double r = Rand[++t];

if (r < 0.25) number = 3;

else if (r < 0.5) number = 4;

else if (r < 0.75) number = 5;

else number = 6;

for (int j = 0; j < number; j++)

{

if (Rand[++t] < 0.5)

{

is\_precise = false;

cur\_time += standard\_time(new double[2] { Rand[++t], Rand[++t] });

}

else cur\_time += standard\_time(new double[2] { Rand[++t], Rand[++t] });

}

time += cur\_time;

if (is\_precise) precise\_count++;

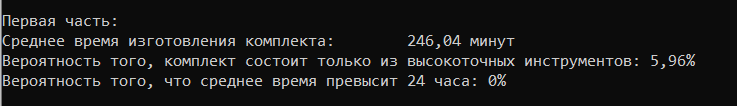
if (cur\_time > value) exceeding++;

count++;

}

}

Результат выполнения этой части программы



### 3.4.1 Часть 1

Для выполнения первой части третьего задания был написан класс Part\_3\_2, вот основной его метод:

public void Run(double[] Rand)

{

int t = 0;

double r\_sum;

double cur\_time;

bool is\_serviceable;

for (int i = 0; i < 2; i++)

{

cur\_time = 0;

is\_serviceable = true;

for (int j = 0; j < 5; j++)

{

cur\_time += 2 + (5 - 2) \* Rand[++t];

if (Rand[++t] <= 0.1)

{

r\_sum = 0;

for (int k = 0; k < 6; k++)

r\_sum += Rand[++t];

cur\_time += 5 + 0.5 \* Math.Sqrt(2) \* (r\_sum - 3);

is\_serviceable = false;

subst\_number++;

}

}

if (is\_serviceable)

cur\_time = -6 \* Math.Log(1 - Rand[++t]);

else

{

cur\_time = -10 \* Math.Log(1 - Rand[++t]);

subst\_count++;

}

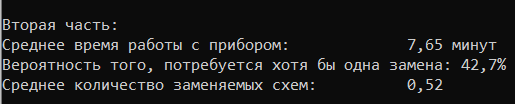
time += cur\_time;

count++;

}

}

Результат выполнения этой части программы:



## Вывод

В ходе выполнения лабораторной работы научились имитировать случайные события методом Монте-Карло.