МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА №52

Отчет защищен с оценкой________________________________

Преподаватель

Доцент, КТН		Н.В.Марковская
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3

по курсу: НАДЕЖНОСТЬ ИНФОКОММУНИКАЦИОННЫХ СИСТЕМ

Студент гр. №	5912		И.К. Лобач
	номер группы	подпись, дата	инициалы, фамилия

Санкт-Петербург 2022

1 Цель работы

Исследование интенсивности отказов и функции надежности для невосстанавливаемых систем путем имитационного моделирования процесса функционирования невосстанавливаемой системы для трех периодов жизни системы. Построение зависимости $\lambda(t)$ оценки интенсивности отказов от времени и функции надежности $R(t)$.

2 Имитационное моделирование процесса функционирования невосстанавливаемых систем

При имитационном моделировании необходимо провести N экспериментов. В каждом эксперименте моделируется процесс функционирования одного экземпляра системы. Для i-ой системы моделирование состоит в вычислении значения случайной величины T_i – времени работы этой системы до момента отказа. Для каждого периода жизни системы случайная величина T_i вычисляется по своему алгоритму.

3 Исходные данные

Моделирование осуществляется для N = 30 000 систем, которые поделены на $k = 2$ групп, причем вероятность попадания в первую группу $p_1 = 0.4$, а во вторую $p_2 = 0.6$. Интенсивность отказа для первой системы $\lambda_1 = 0.9$ и для второй $\lambda_2 = 1.1$.

4 Моделирование первого периода

Для i-ой системы первоначально надо определить, к какому подмножеству она относится. Для этого используется распределение p_i. После того, как номер i подмножества определен, необходимо сгенерировать значение T_i, как случайной величины, распределенной по экспоненциальному закону с параметром λ_i, который определяется по номеру подмножества.

Определение T_i осуществляется по формуле:

$$T_i = \frac{-\ln{[0,1]}}{\lambda_i}$$

Среднее время работы i-ой системы определяется по формуле:

$$\overline{T} = \sum_{i=0}^{n} T_i$$

Затем для каждого t до $\overline{T}$ с шагом $\Delta t = 0.01$ происходит подсчет систем, которые работают.

Теоретическое значение функции надежности:

$$R(t) = R_1(t)p_1 + R_2(t)p_2 = e^{-\lambda_1 t}p_1 + e^{-\lambda_2 t}p_2$$

Экспериментальное значение функции надежности:

$$\hat{R}(t) = \frac{n_t}{n}$$

где, $n_t -$ число систем, работающих в момент времени t, $n -$ общее число систем.

Графики:

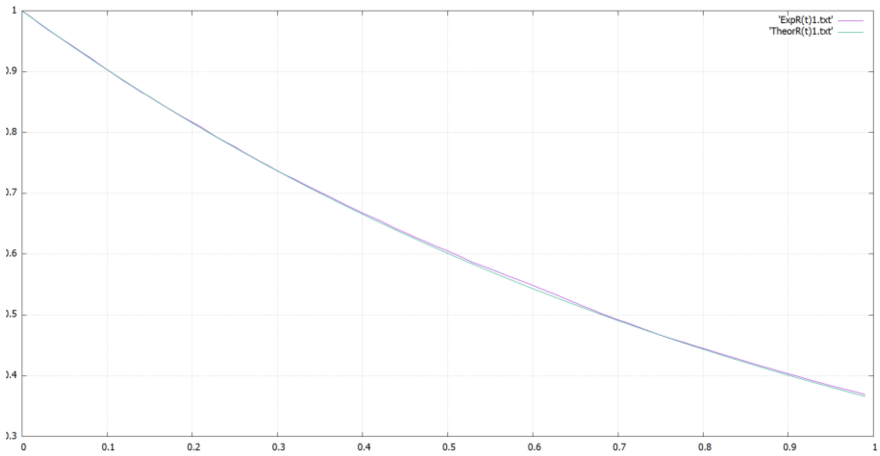

График 1 - Функция надежности для первого периода

Теоретическое значение интенсивности отказа:

$$\lambda(t) = -\frac{R'(t)}{R(t)} = -\frac{(-\lambda_1 p_1)e^{-\lambda_1 t} + (-\lambda_2 p_1)e^{-\lambda_2 t}}{e^{-\lambda_1 t}p_1 + e^{-\lambda_2 t}p_2}$$

Экспериментальное значение интенсивности отказа:

$$\hat{\lambda}(t) = \frac{n_t - n_{t+\Delta t}}{n_t \Delta t}$$

где $n_t -$ число работоспособных систем в момент t, $n_{t+\Delta t} -$ число систем, работающих в момент $t + \Delta t$, где $\Delta t = 0.001$.

Графики:

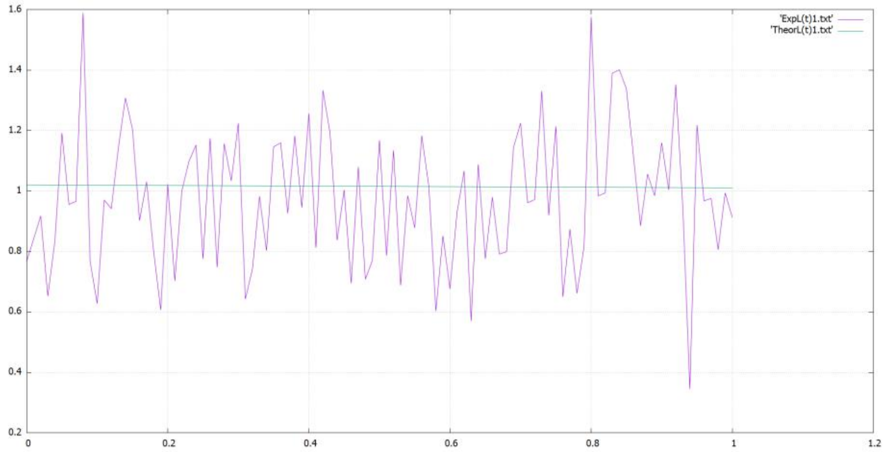

График 2 - Интенсивность отказа для первого периода

5 Моделирование второго периода

Для i-ой системы первоначально надо определить, к какому подмножеству она относится. Для этого используется распределение p_i. После того, как номер i подмножества определен, необходимо сгенерировать значение T_i, как случайной величины, распределенной по экспоненциальному закону с параметром λ_i, который определяется по номеру подмножества. Значение времени i-ой системы определяется как

$$T = \min T_i$$

Среднее время работы i-ой системы определяется по формуле:

$$\overline{T} = \sum_{i=0}^{n} T_i$$

Затем для каждого t до $\overline{T}$ с шагом $\Delta t = 0.01$ происходит подсчет систем, которые работают.

Теоретическое значение функции надежности:

$$R(t) = R_1(t)R_2(t) = e^{-(\lambda_1+\lambda_2)t}$$

Экспериментальное значение функции надежности:

$$\hat{R}(t) = \frac{n_t}{n}$$

где, n_t – число систем, работающих в момент времени t, n – общее число систем.

Графики:

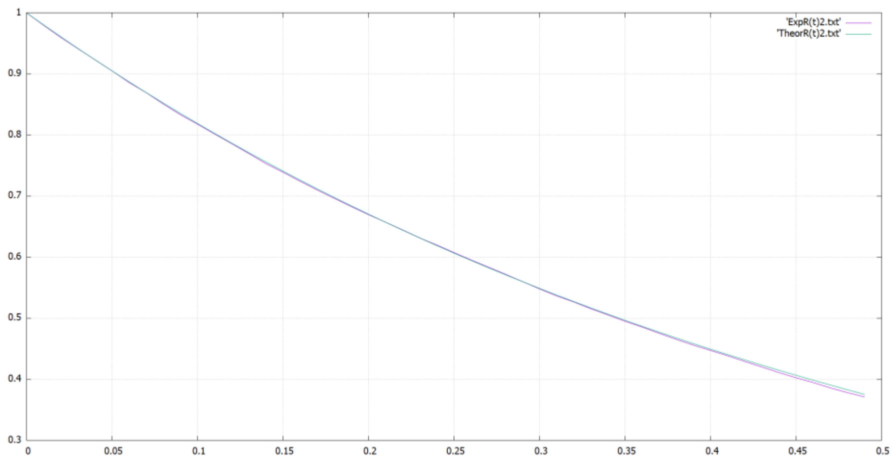

График 3 - Функция надежности для второго периода

Теоретическое значение интенсивности отказа:

$$\lambda(t) = -\frac{R'(t)}{R(t)} = -\frac{-(\lambda_1 + \lambda_2)e^{-(\lambda_1+\lambda_2)t}}{e^{-\lambda_1 t}e^{-\lambda_2 t}} = \lambda_1 + \lambda_2$$

Экспериментальное значение интенсивности отказа:

$$\hat{\lambda}(t) = \frac{n_t - n_{t+\Delta t}}{n_t \Delta t}$$

где $n_t -$ число работоспособных систем в момент t, $n_{t+\Delta t} -$число систем, работающих в момент $t + \Delta t$, где $\Delta t = 0.001$.

Графики:

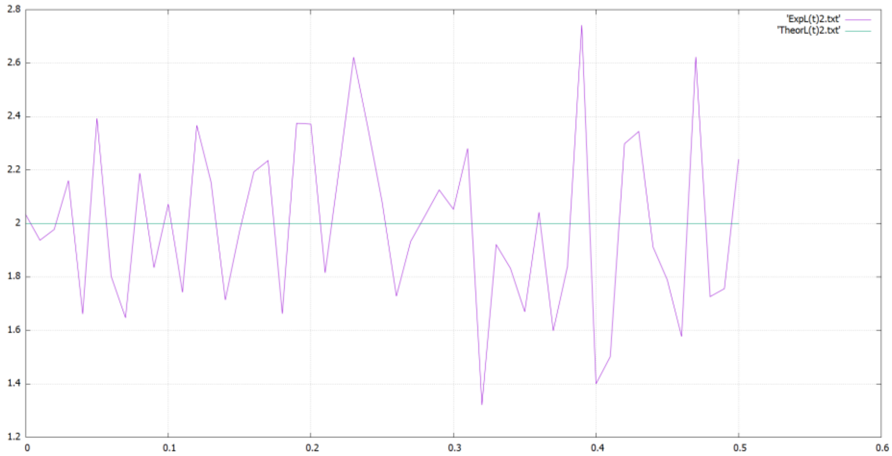

График 4 - Интенсивность отказа для второго периода

6 Моделирование третьего периода

Для i-ой системы первоначально надо определить, к какому подмножеству она относится. Для этого используется распределение p_i. После того, как номер i подмножества определен, необходимо сгенерировать значение T_i, как случайной величины, распределенной по экспоненциальному закону с параметром λ_i, который определяется по номеру подмножества. Значение времени i-ой системы определяется как

$$T = \max T_i$$

Среднее время работы i-ой системы определяется по формуле:

$$\overline{T} = \sum_{i=0}^{n} T_i$$

Затем для каждого t до $\overline{T}$ с шагом $\Delta t = 0.01$ происходит подсчет систем, которые работают.

Теоретическое значение функции надежности:

$$R(t) = R_1(t) + R_2(t) = e^{-\lambda_1 t} + e^{-\lambda_2 t} - e^{-(\lambda_1+\lambda_2)t}$$

Экспериментальное значение функции надежности:

$$\hat{R}(t) = \frac{n_t}{n}$$

где, n_t – число систем, работающих в момент времени t, n – общее число систем.

Графики:

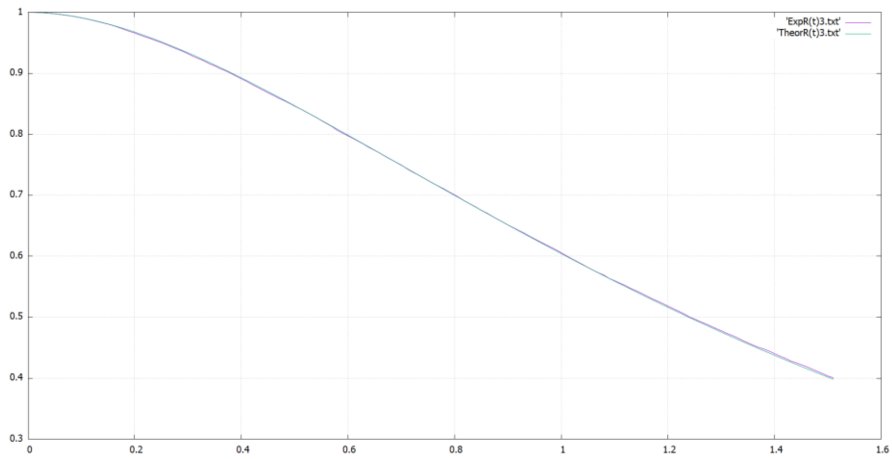

График 5 - Функция надежности для третьего периода

Теоретическое значение интенсивности отказа:

$$\lambda(t) = -\frac{R'(t)}{R(t)} = -\frac{(-\lambda_1)e^{-\lambda_1 t} + (-\lambda_2)e^{-\lambda_2 t} - (\lambda_1 + \lambda_2)e^{-(\lambda_1+\lambda_2)t}}{e^{-\lambda_1 t} + e^{-\lambda_2 t} - e^{-(\lambda_1+\lambda_2)t}}$$

Экспериментальное значение интенсивности отказа:

$$\hat{\lambda}(t) = \frac{n_t - n_{t+\Delta t}}{n_t \Delta t}$$

где n_t – число работоспособных систем в момент t, $n_{t+\Delta t}$ – число систем, работающих в момент $t + \Delta t$, где $\Delta t = 0.001$.

Графики:

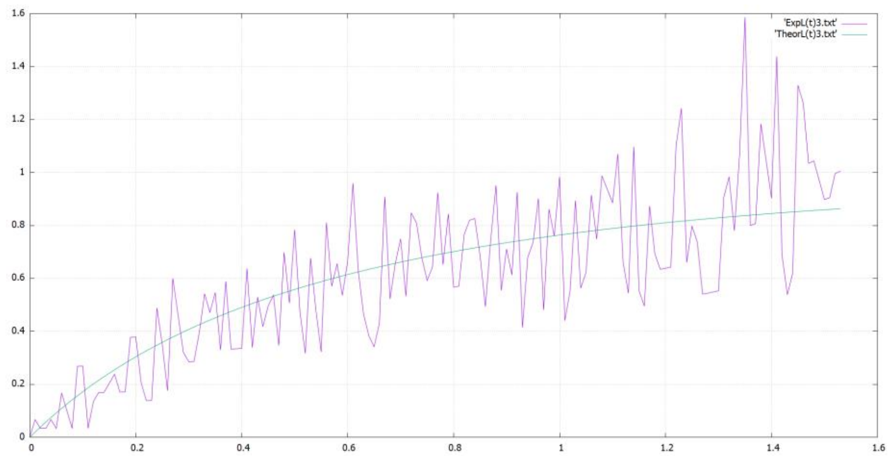

График 6 - Интенсивность отказа для третьего периода

7 Выводы

Таким образом, в ходе моделирования трех периодов работы невосстанавливаемых систем, были получены экспериментальные значения функции надежности $R(t)$ и интенсивности отказа $\lambda(t)$.

Экспериментальные и теоретические значения близки по значения, что может говорить о корректной работе программы.

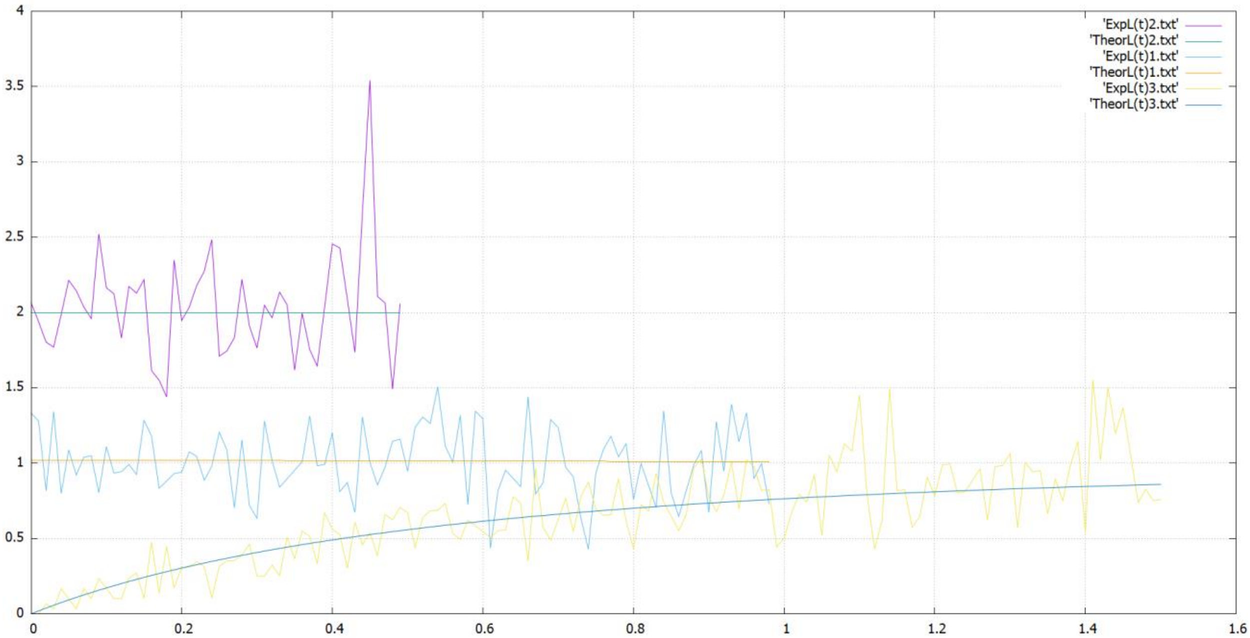

График 7 - Сравнение интенсивности отказа

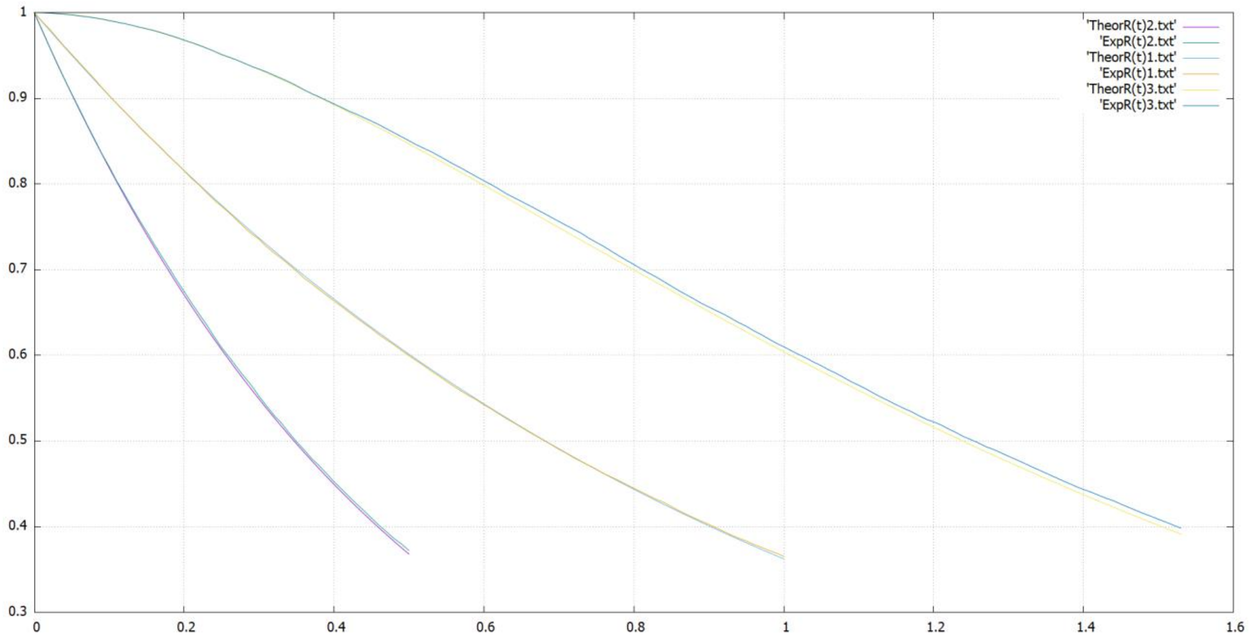

График 8 - Сравнение функции надежности

<u>Листинг программы</u>

```
package com.suai;

import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;

public class Modeling {

 private int n = 30000;
 private double l1 = 0.9;
 private double l2 = 1.1;
 private double p1 = 0.4;
 private double p2 = 0.6;
 private double T;
 private ArrayList Ti = new ArrayList();


 private void timeModeling1() {
  for (int i = 0; i < n; i++) {
   double tmp = Math.random();
   if (((double) i / n - p1) < 0.01) {
    Ti.add((Math.log(tmp) / l1) * (-1));
   } else {
    Ti.add((Math.log(tmp) / l2) * (-1));
   }
  }
 }

 private void timeModeling2() {
  for (int i = 0; i < n; i++) {
   double tmp1 = (-1)*Math.log(Math.random());
   double tmp2 = (-1)*Math.log(Math.random());
   if (tmp1/l1 <= tmp2/l2)
    Ti.add(tmp1/l1);
   else
    Ti.add(tmp2/l2);
  }
 }
 private void timeModeling3() {
  for (int i = 0; i < n; i++) {
   double tmp1 = (-1)*Math.log(Math.random());
   double tmp2 = (-1)*Math.log(Math.random());
   if (tmp1/l1 >= tmp2/l2)
    Ti.add(tmp1/l1);
   else
    Ti.add(tmp2/l2);
```

```
  }
 }

 public void getAverageT() {
  for (int i = 0; i < Ti.size(); i++) {
    T += (double) Ti.get(i);
  }
  T /= n;
 }

 public void getTheoretical1() {
  for (double t = 0; t - T <= 0.0; t += 0.01) {
    double R = 0;
    double l = 0;
    R = Math.exp(-l1*t)*p1 + Math.exp(-l2*t)*p2;
    l = (l1*p1*Math.exp((-1)*l1*t) + l2*p2*Math.exp((-1)*l2*t))/R;
    writeToFile(R,t, "TheorR(t)1.txt");
    writeToFile(l,t, "TheorL(t)1.txt");
  }
 }

 public void getTheoretical2() {
  double l = l1 + l2;
  for (double t = 0; t - T <= 0.0; t += 0.01) {
    double R = 0;
    R = Math.exp(-l1*t) * Math.exp(-l2*t);
    writeToFile(R,t, "TheorR(t)2.txt");
    writeToFile(l,t, "TheorL(t)2.txt");
  }
 }

 public void getTheoretical3() {
  for (double t = 0; t - T <= 0.0; t += 0.01) {
    double R = 0;
    double l = 0;
    R = Math.exp(-l1*t) + Math.exp(-l2*t) - Math.exp(-l1*t)*Math.exp(-l2*t);
    writeToFile(R,t, "TheorR(t)3.txt");
    l = (-1)*((-1)*l1*Math.exp((-1)*l1*t) - l2*Math.exp((-1)*l2*t) + (l1+l2)*Math.exp((-
1)*(l1+l2)*t))/R;
    writeToFile(l,t, "TheorL(t)3.txt");
  }
 }

 public int findN(double t) {
  int nT = 0;
  for (int i = 0; i < Ti.size(); i++) {
    if (t <= (double)Ti.get(i)) {
      nT++;
    }
```

```
  }
  return nT;
}

public void getExperimental(String filename1, String filename2) {
  for (double t = 0; t < T; t += 0.01) {
    int nT = findN(t);
    double R = (double)nT/n;
    writeToFile(R,t, filename1);
    double l = (double)(nT - findN(t+0.001))/ (nT*0.001);
    writeToFile(l,t, filename2);
  }
}

public void writeToFile(double R, double t, String filename) {
  try {
    FileWriter file = new FileWriter(filename, true);
    StringBuilder str = new StringBuilder();
    str.append(t).append(" ").append(R).append("\n");
    file.write(str.toString());
    file.flush();
  } catch (IOException ex) {
    System.out.println(ex.getMessage());
    ex.printStackTrace();
  }
}

public void firstPeriodModeling() {
  Ti.clear();
  T = 0;
  timeModeling1();
  getAverageT();
  getExperimental("ExpR(t)1.txt", "ExpL(t)1.txt");
  getTheoretical1();
}

public void secondPeriodModeling() {
  Ti.clear();
  T = 0;
  timeModeling2();
  getAverageT();
  getExperimental("ExpR(t)2.txt", "ExpL(t)2.txt");
  getTheoretical2();
}
public void thirdPeriodModeling() {
  Ti.clear();
  T = 0;
  timeModeling3();
  getAverageT();
```

```
    getExperimental("ExpR(t)3.txt", "ExpL(t)3.txt");
    getTheoretical3();
  }

  public void clearFile() {
    try {
      String[] files = new String[12];
      files[0] = "TheorR(t)1.txt";
      files[1] = "TheorL(t)1.txt";
      files[2] = "ExpR(t)1.txt";
      files[3] = "ExpL(t)1.txt";
      files[4] = "TheorR(t)2.txt";
      files[5] = "TheorL(t)2.txt";
      files[6] = "ExpR(t)2.txt";
      files[7] = "ExpL(t)2.txt";
      files[8] = "TheorR(t)3.txt";
      files[9] = "TheorL(t)3.txt";
      files[10] = "ExpR(t)3.txt";
      files[11] = "ExpL(t)3.txt";
      for (int i = 0; i < files.length; i++) {
        PrintWriter file = new PrintWriter(files[i]);
        file.close();
      }
    }
    catch(IOException e) {
      System.out.println(e.getMessage());
      e.printStackTrace();
    }
  }

  public static void main(String[] args) {
    Modeling m = new Modeling();
    m.firstPeriodModeling();
    m.secondPeriodModeling();
    m.thirdPeriodModeling();
  }

}
```