**Министерство науки и высшего образования РФ**

ФГАОУ ВО «Уральский федеральный университет

имени первого Президента России Б.Н. Ельцина»

ИРИТ - РТФ

Центр ускоренного обучения

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2**

по дисциплине «Проектирование информационных систем»

|  |  |
| --- | --- |
| Студенты группы РИВ-420906у: | С.Г. Миронов  А.В. Литвиненко  А.Д. Бирючев  Д.А. Махонин  А.В. Васильев |
| Преподаватель: | С.И. Тимошенко доц., к.т.н. |

Екатеринбург 2025

# СОДЕРЖАНИЕ

СОДЕРЖАНИЕ 2

1 Постановка задачи 3

1.1 3

1.2 3

1.3 3

1.4 3

2 2 Анализ поставленной задачи 4

2.1 4

2.2 4

2.3 4

3 Описание результатов 5

3.1 5

3.2 5

3.3 5

3.4 5

4 Анализ полученных результатов 6

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 7

ПРИЛОЖЕНИЕ А 8

# 1 Постановка задачи

Разработать программное обеспечение, выполняющее расчеты при любых заданных входных параметрах для модели Джелинского-Моранды. В качестве тестовых использовать входные параметры, приведенные ниже в описании модели. Необходимо рассчитать параметры, описанные ниже в п. “Порядок расчета”.

Модель Джелинского-Моранды – одна из первых и наиболее простых моделей классического типа. Модель использовалась при разработке ПО для весьма ответственных проектов, в частности для ряда модулей программы Apollo. В ее основу были положены следующие допущения:

* функция риска или иначе – интенсивность обнаружения ошибок R(t) пропорциональна текущему числу ошибок в программе, т.е. числу оставшихся (первоначальных) ошибок минус обнаруженные;
* все ошибки одинаково вероятны и их появление не зависит друг от друга;
* каждая ошибка имеет один и тот же порядок серьезности;
* время до следующего отказа распределено экспоненциально;
* ПО функционирует в среде, близкой к реальным условиям;
* ошибки постоянно корректируются без внесения в ПО новых;
* R(t)=const в интервале между двумя смежными моментами появления ошибок.

В соответствии с этими допущениями можно функцию риска представить в следующем виде:

R(t)=K[B-(i-1)],

где t – произвольная точка времени между обнаружением (i-1)-й и i-й ошибок;

К – неизвестный коэффициент пропорциональности;

B – исходное (неизвестное) общее число оставшихся в ПО ошибок.

Таким образом, если за время t было обнаружено (i-1) ошибок, то в ПО еще остается   
B-(i-1) ошибок. Полагая Xi=ti-ti-1 (i=1,n) и используя допущение 7, а также формулу функции риска, можно утверждать, что все Xi имеют экспоненциальное распределение:

P(Xi)=exp{-K[B-(i-1)]Xi},

а плотность вероятности отказов соответственно равна

q(Xi)=K[B-(i-1)]exp{-K[B-(i-1)]Xi}.

Тогда функция правдоподобия (согласно допущению 2 будет равна

L(X1,…,Xn)=, (1)

или, перейдя к логарифму функции правдоподобия, получим

L(X1,…,Xn)=[ln(K(B-i+1))-K(B-i+1)Xi]. (2)

Условия для нахождения экстремума (2) следующие:

, (3)

. (4)

Из (3) можно получить оценку максимального правдоподобия для К, а подставив его в (4) найти нелинейное уравнение для оценки максимального правдоподобия величины B.

Ниже предлагается решить задачу оценки числа ошибок в программной системе, используя вышеприведенные сведения о модели Джелинского-Моранды. Приведен также примерный поря­док расчета, в том числе конечные выражения для B и K, полученные из системы уравнений (3), (4).

Входные параметры.

В таблицах ниже приведены данные тестирования и отладки критической программной системы по вариантам (по номерам команд). Найти общее число ошибок в программной системе, время до появления следующей ошибки и время до окончания тестирования. Для расчетов использовать модель Джелинского-Моранды.

Таблица 1 – Хронология тестирования. Вариант 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Номер ошибки  (i) | Интервал  между ошибками  (Xi, часы) | Номер ошибки  (i) | Интервал  между ошибками  (Xi, часы) | Номер ошибки  (i) | Интервал  между ошибками  (Xi, часы) | Номер ошибки  (i) | Интервал  между ошибками  (Xi, часы) |
| 1 | 9 | 8 | 8 | 15 | 4 | 22 | 33 |
| 2 | 12 | 9 | 5 | 16 | 1 | 23 | 7 |
| 3 | 11 | 10 | 7 | 17 | 3 | 24 | 91 |
| 4 | 4 | 11 | 1 | 18 | 3 | 25 | 2 |
| 5 | 7 | 12 | 6 | 19 | 6 | 26 | 1 |
| 6 | 2 | 13 | 1 | 20 | 1 |  |  |
| 7 | 5 | 14 | 9 | 21 | 11 |  |  |

Порядок расчета.

1. Решить численным методом следующее нелинейное уравнение для оценки максимального правдоподобия величины B (общее число ошибок в программе):

,

где n – число обнаруженных ошибок;

Xi – интервал времени до появления i-й ошибки.

1. Найти коэффициент пропорциональности K (используется в функции риска R(t)=K[B-(i-1)], где t –произвольная точка времени между обнаружением (i-1)-й и i-й ошибок):

.

1. Найти среднее время Xn+1 до появления (n+1)-й ошибки как величину, обратную оценочной интенсивности для предыдущей ошибки:

.

1. Найти время до окончания тестирования:



# 2 2 Анализ поставленной задачи

## 2.1

## 2.2

## 2.3

# 3 Описание результатов

## 3.1

## 3.2

## 3.3 Диаграмма классов

По результатам выполнения лабораторной работы была создана диаграмма классов, которую можно увидеть на рисунке 1.

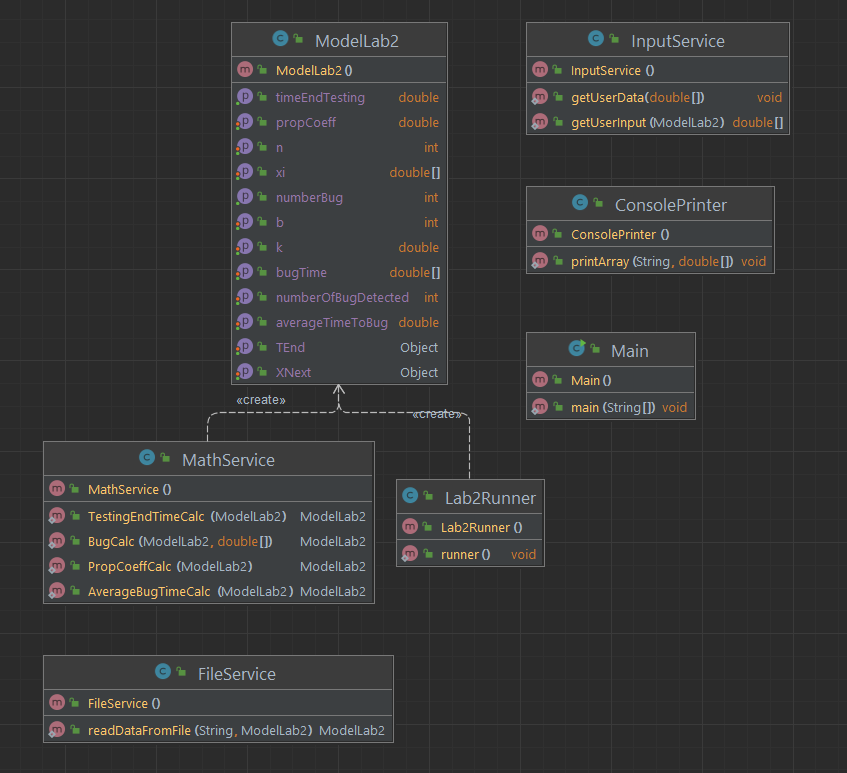


Рисунок 1 – Результат построения диаграммы классов

## 3.4 Описание работы классов

Далее рассмотрим более подробно каждый класс созданный в процессе выполнения лабораторной работы:

1. ModelLab2

* Описание: Этот класс представляет модель данных, используемую для хранения информации о тестировании.
* Атрибуты:

1. numberOfBugDetected: хранит количество обнаруженных ошибок;
2. bugTime[]: массив, содержащий время появления ошибок;
3. numberBug: общее количество ошибок;
4. propCoeff: коэффициент пропорциональности;
5. averageTimeToBug: среднее время до появления ошибки;
6. timeEndTesting: время до окончания тестирования.

* Методы:

1. Геттеры и сеттеры для управления значениями атрибутов;
2. Методы setN, setXi, setB, setK, setXNext и setTEnd предназначены для установки значений атрибутов.
3. FileService

* Описание: Этот класс отвечает за чтение данных из файла.
* Методы:

1. readDataFromFile(String filename, ModelLab2 model): читает данные из указанного файла, парсит их и обновляет объект ModelLab2 с полученными данными.
2. InputService

* Описание: Класс предназначен для взаимодействия с пользователем и получения входных данных.
* Методы:

1. getUserInput(ModelLab2 model): запрашивает у пользователя верхнюю и нижнюю границы интервала и погрешность, проверяет введенные данные и обновляет объект ModelLab2;
2. getUserData(double[] userData): получает дополнительные данные от пользователя, такие как нижняя граница интервала и погрешность.
3. MathService

* Описание: Класс, который выполняет математические расчеты, связанные с моделью тестирования.
* Методы:

1. BugCalc(ModelLab2 model, double[] userData): оценивает общее количество ошибок B с помощью метода бисекции;
2. PropCoeffCalc(ModelLab2 model): рассчитывает коэффициент пропорциональности K;
3. AverageBugTimeCalc(ModelLab2 model): оценивает время до следующей ошибки (X\_{n+1});
4. TestingEndTimeCalc(ModelLab2 model): вычисляет время до окончания тестирования на основе предыдущих результатов.
5. Lab2Runner

* Описание: Этот класс запускает выполнение лабораторной работы, инициализируя модель и последовательно выполняя расчеты.
* Методы:

1. runner(): создает объект ModelLab2, считывает входные данные, выполняет математические расчеты и выводит результаты на консоль.
2. ConsolePrinter

* Описание: Этот класс отвечает за вывод данных на консоль.
* Методы:

1. printArray(String message, double[] array): выводит указательное сообщение и содержимое массива double[]. Обрабатывает случаи, когда массив пуст или равен null.
2. Main

* Описание: Основной класс приложения.
* Методы:

1. main(String[] args): точка входа в программу. Вызывает метод runner() из класса Lab2Runner для выполнения теста.

# 4 Анализ полученных результатов

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

# ПРИЛОЖЕНИЕ А

Listing 1 ­–