Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное
образовательное учреждение высшего образования
«Самарский национальный исследовательский университет имени академика С.П. Королева»

Институт информатики и кибернетики

Кафедра технической кибернетики

Отчет по лабораторной работе №2

Дисциплина: «ООП»

Тема «Разработать набор классов для работы с функциями одной переменной, заданными в табличной форме.»

Выполнил: Зорин Дмитрий Сергеевич

Группа: 6201-120303D

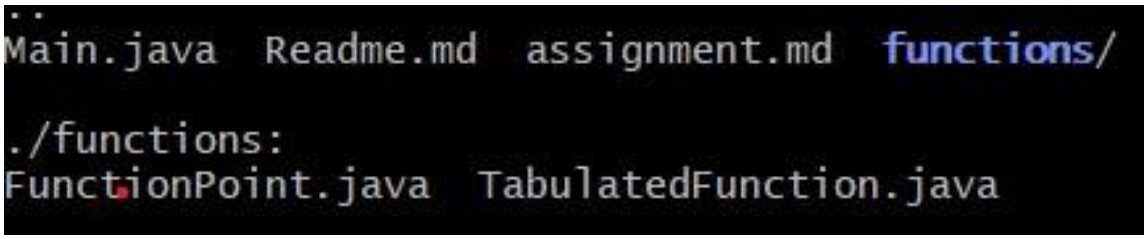
Самара, 2025

Задание на лабораторную работу

Задание 1

Создать пакет functions, в котором далее будут создаваться классы программы

Структура проекта:



Задание 2

В пакете functions создал класс FunctionPoint, объект которого должен описывать одну точку табулированной функции **Реализовано:**
Конструкторы: FunctionPoint(double x, double y), FunctionPoint(FunctionPoint point), FunctionPoint()

Методы доступа и изменения координат: getX(), getY(), setX(), setY()

Результат:
Объекты класса корректно создаются и могут использоваться для формирования массива точек табулированной функции

Задание 3

В пакете functions создал класс TabulatedFunction, объект которого должен описывать табулированную функцию
В классе описал данные конструкторы:

TabulatedFunction(double leftX, double rightX, int pointsCount) – создаёт функцию с pointsCount

TabulatedFunction(double leftX, double rightX, double[] values)

Вычисляем шаг step = (rightX - leftX) / (pointsCount - 1).

Задание 4

В классе TabulatedFunction описал методы, необходимые для работы с функцией

Методы для работы с функцией:

getLeftDomainBorder() – возвращает левую границу области

getRightDomainBorder() – возвращает правую границу области определения

getFunctionValue(double x) – возвращает значение функции в точке x

Алгоритм getFunctionValue:

Если x вне области [leftX, rightX], возвращается Double.NaN

Если x совпадает с точкой массива, возвращается её y.

Задание 5

В классе TabulatedFunction описал методы, необходимые для работы с точками табулированной функции **Методы для работы с точками:**

getPointsCount() – возвращает количество точек getPoint(int

index)

setPoint(int index, FunctionPoint point) – заменяет точку, если x нового объекта находится между соседними точками.

getPointX(int index) / setPointX(int index, double x) getPointY(int

index) / setPointY(int index, double y)

Методы необходимы, чтобы изменение координаты x или замена точки не нарушает порядок точек по x.

Задание 6

В классе TabulatedFunction описал методы, изменяющие количество точек

табулированной функции **Добавление и удаление точек:**

addPoint(FunctionPoint point) deletePoint(int index) – удаление точки

Добавление:

Создаём новый массив на 1 элемент больше

Копируем элементы до insertIndex, вставляем новую точку, копируем остаток массива

Удаление:

Создаём новый массив на 1 элемент меньше

Копируем элементы до удаляемой точки, пропускаем её, копируем остаток

Задание 7

Проверил работу написанных классов

Создал класс Main вне пакета functions

Создан экземпляр TabulatedFunction

Проверил работу методов добавления и удаления точек, а также изменения координат

Проверил значения функции для нескольких x, включая точки вне области и внутри интервалов.

Результаты выполнения программы :

Исходная функция:

(0.0, 0.0)

(1.0, 1.0)

(2.0, 4.0)

(3.0, 9.0)

Проверка значений в разных точках:

(-1,00) = NaN

(0,50) = 0,50

(1,50) = 2,50

(2,50) = 6,50

(4,00) = NaN

Добавляем точку (1.5, 2.25):

(0.0, 0.0)

(1.0, 1.0)

(1.5, 2.25)

(2.0, 4.0)

(3.0, 9.0)

Удаляем точку с индексом 2:

(0.0, 0.0)

(1.0, 1.0)

(2.0, 4.0)

(3.0, 9.0)