ЛАБОРАТОРНАЯ РАБОТА №2

Разработка классов для работы с табулированными функциями.

по курсу
Объектно-ориентированное программирование

Группа 6204-010302D

Студент: С.0.Куропаткин.

Преподаватель: Борисов Дмитрий

Сергеевич.

Задание 1: Создание пакета functions

Ход выполнения: Создан пакет functions для организации структуры проекта и логической группировки связанных классов.

Результат: Пакет functions создан и содержит классы FunctionPoint и TabulatedFunction.

Задание 2: Класс FunctionPoint

Ход выполнения: Реализован класс FunctionPoint, представляющий точку с координатами (x, y). Объявлены поля private x и private y.

Конструкторы:

- FunctionPoint(double x, double y) - точка с заданными координатами
- FunctionPoint(FunctionPoint point) - конструктор копирования
- FunctionPoint() - конструктор по умолчанию

Результат: Класс успешно создан.

Задание 3: Класс TabulatedFunction - конструкторы

Ход выполнения: Создан класс для работы с табулированными функциями. Точки хранятся в массиве, упорядоченном по координате x.

Конструкторы:

- TabulatedFunction(double leftX, double rightX, int pointsCount) - значения функции равны 0.
- TabulatedFunction(double leftX, double rightX, double[] values) - заданные значения функции

Результат: Конструкторы корректно создают объекты табулированных функций.

Задание 4: Методы работы с функцией

Ход выполнения: Реализованы методы для работы с областью определения и вычисления значений функции.

Методы:

- getLeftDomainBorder() - левая граница области определения
- getRightDomainBorder() - правая граница области определения

- getFunctionValue(double x) - значение функции в точке (линейная интерполяция)

Результат: Методы корректно работают, включая обработку точек вне области определения.

Задание 5: Методы работы с точками

Ход выполнения: Реализованы методы для доступа и модификации точек функции.

Методы:

- getPointsCount() - количество точек
- getPoint(int index) - копия точки (инкапсуляция)
- setPoint(int index, FunctionPoint point) - замена точки
- Геттеры и сеттеры для координат x и y

Результат: Обеспечен безопасный доступ и модификация точек с сохранением порядка.

Задание 6: Изменение количества точек

Ход выполнения: Реализованы методы для изменения структуры табулированной функции.

Методы:

- deletePoint(int index) - удаление точки
- addPoint(FunctionPoint point) - добавление точки

Результат: Методы эффективно работают с оптимизацией использования памяти.

Задание 7: Тестирование

Ход выполнения: Создан класс Main для тестирования функциональности.

****Тестовый сценарий:****

1. Создание квадратичной функции
2. Проверка значений в различных точках
3. Тестирование интерполяции
4. Проверка операций с точками

Вывод значений в консоль:

x = 0.0, y = 0.0
x = 2.0, y = 1.0
x = 4.0, y = 4.0
x = 6.0, y = 9.0
x = 8.0, y = 16.0
x = 10.0, y = 25.0

Проверка значений функции в различных точках

f(-1.0) = NaN
f(0.0) = 0.0
f(3.5) = 3.25
f(5.0) = 6.5
f(8.5) = 18.25
f(10.0) = 25.0
f(12.0) = NaN

Замена точки с индексом 1 на точку (2,10)

x = 0.0, y = 0.0
x = 2.0, y = 10.0
x = 4.0, y = 4.0
x = 6.0, y = 9.0
x = 8.0, y = 16.0
x = 10.0, y = 25.0

Добавляем новую точку (11,30)

x = 0.0, y = 0.0
x = 2.0, y = 10.0
x = 4.0, y = 4.0
x = 6.0, y = 9.0
x = 8.0, y = 16.0
x = 10.0, y = 25.0
x = 11.0, y = 30.0

Удаляем точку с индексом 2

x = 0.0, y = 0.0
x = 2.0, y = 10.0
x = 6.0, y = 9.0
x = 8.0, y = 16.0
x = 10.0, y = 25.0
x = 11.0, y = 30.0