

Лабораторная работа №4

Расширение пакета функций: интерфейсы, базовые функции, табулирование и ввод/вывод

Студент: Стоколяс Юрий Юрьевич

Группа: 6201-120303D

Задание 1: Конструктор из массива точек

В `functions.TabulatedFunction` добавлен конструктор, принимающий массив `FunctionPoint[]`. Выполняется проверка, что точек не меньше двух и абсциссы строго возрастают. При нарушении выбрасывается `IllegalArgumentException`. Точки копируются для соблюдения инкапсуляции.

Задание 2: Интерфейс Function

Создан интерфейс `functions.Function` с методами:

- `double getLeftDomainBorder()`
- `double getRightDomainBorder()`
- `double getFunctionValue(double x)`

`TabulatedFunction` реализует `Function`.

Реализация TabulatedFunction

Класс хранит точки в массиве `FunctionPoint[]`, поддерживает упорядоченность по `x`, конструкторы:

- `TabulatedFunction(double leftX, double rightX, int pointsCount)`
- `TabulatedFunction(double leftX, double rightX, double[] values)`

- `TabulatedFunction(FunctionPoint[] points)`

Реализованы вычисление по линейной интерполяции, доступ/модификация точек, добавление/удаление, автоматическое расширение массива.

Задание 3: Базовые функции (аналитические)

Добавлены классы в `functions.basic`:

- `Sin` — синус
- `Cos` — косинус
- `Exp` — экспонента
- `Log` — натуральный логарифм (и по основанию при передаче параметра конструктора)

Все реализуют `Function`.

Задания 4–7: Табулирование, операции и ввод/вывод

Добавлен `functions.Functions` с операциями:

- `sum(a, b)` — сумма функций
- `mult(a, b)` — произведение функций
- `power(f, p)` — возведение значения функции в степень `p`

Добавлен `functions.TabulatedFunctions`:

- `tabulate(Function f, left, right, pointsCount)` — табулирование
 - `outputTabulatedFunction/inputTabulatedFunction` — бинарный формат
 - `writeTabulatedFunction/readTabulatedFunction` — текстовый формат
-

Задание 8: Тестирование

- Табулированная `x^2` (создание, чтение значений, добавление/удаление точки)

- Значения `Sin/Cos` и их табулированные аналоги; сумма квадратов табулированных значений
- Табулирование `Exp` с текстовым I/O, `Log` с бинарным I/O

Задание 9: Сериализация

`functions.TabulatedFunction` реализует `Externalizable` (методы `writeExternal/readExternal`). В `Main` добавлен тест сериализации объекта табулированной функции с последующей десериализацией и сравнением значений на отрезке 0..10 с шагом 1.

Результаты запуска

Функция `x^2`:

Область определения: `[0.0, 4.0]`

Количество точек: `5`

Значения функции:

`f(-1.0)` = не определено

`f(0.0)` = `0.0`

`f(0.5)` = `0.5`

`f(1.0)` = `1.0`

`f(1.5)` = `2.5`

`f(2.0)` = `4.0`

`f(2.5)` = `6.5`

`f(3.0)` = `9.0`

`f(3.5)` = `12.5`

`f(4.0)` = `16.0`

`f(5.0)` = не определено

Добавляем точку `(1.5, 2.25)`:

Количество точек после добавления: `6`

Удаляем точку с индексом `2`:

Количество точек после удаления: `5`

Значения функции после изменений:

$f(-1.0)$ = не определено

$f(0.0)$ = 0.0

$f(0.5)$ = 0.5

$f(1.0)$ = 1.0

$f(1.5)$ = 2.5

$f(2.0)$ = 4.0

$f(2.5)$ = 6.5

$f(3.0)$ = 9.0

$f(3.5)$ = 12.5

$f(4.0)$ = 16.0

$f(5.0)$ = не определено

Информация о точках:

Точка 0: (0.0, 0.0)

Точка 1: (1.0, 1.0)

Точка 2: (2.0, 4.0)

Точка 3: (3.0, 9.0)

Точка 4: (4.0, 16.0)

Sin/Cos 0..pi шаг 0.1:

x=0.0: sin=0.0, cos=1.0

x=0.1: sin=0.09983341664682815, cos=0.9950041652780258

x=0.2: sin=0.19866933079506122, cos=0.9800665778412416

...

x=3.1000000000000014: sin=0.04158066243328916, cos=-0.999135150273279

Табулированные sin/cos 0..π шаг 0.1:

x=0.0: tsin=0.0, tcos=1.0

x=0.1: tsin=0.09798155360510165, tcos=0.9827232084876878

x=0.2: tsin=0.1959631072102033, tcos=0.9654464169753757

...

x=3.1000000000000014: tsin=0.04075312817286608, tcos=-0.9928141239548

Сумма квадратов tab(sin)^2+tab(cos)^2 0..π шаг 0.1:

x=0.0: 1.0

x=0.1: 0.9753452893472049

...

x=3.10000000000000014: 0.9873407021801173

exp 0..**10** шаг 1 (текстовый ввод/вывод):

x=0.0: src=1.0, in=1.0

x=1.0: src=2.718281828459045, in=2.718281828459045

...

x=10.0: src=22026.465794806718, in=22026.465794806718

ln 0..**10** шаг 1 (бинарный ввод/вывод):

x=0.0: src=NaN, in=0.0

x=1.0: src=NaN, in=0.0

...

x=10.0: src=2.302585092994046, in=2.302585092994046