

Лабораторная работа №2

Студент: Гонеев Роман Олегович

Группа: 6201-120303D

Задания:

1. Создаю пакет functions

```
#mkdir functions
```

2. Создан класс FunctionPoint.java в пакете functions(в начале класса прописываю package functions, тем самым создаю пакет functions).

```
#nano functions/FunctionPoint.java
```

В классе описываю конструкторы:

- FunctionPoint(double x, double y) – создаёт объект точки с заданными координатами;
- FunctionPoint(FunctionPoint point) – создаёт объект точки с теми же координатами, что у указанной точки;
- FunctionPoint() – создаёт точку с координатами (0; 0).

3. Создан класс TabulatedFunction в пакете functions.

```
#nano functions/TabulatedFunction.java
```

```
TabulatedFunction(double leftX, double rightX, int pointsCount)
```

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

4. В классе TabulatedFunction описаны методы:

```
# nano functions/TabulatedFunction.java
```

```
double getLeftDomainBorder()
```

```
double getRightDomainBorder()
```

```
double getFunctionValue(double x)
```

5. В классе TabulatedFunction описаны методы, необходимые для работы с точками табулированной функции:

```
# nano functions/TabulatedFunction.java
```

```
int getPointsCount()
```

```
void setPoint(int index, FunctionPoint point)
```

```
double getPointX(int index)
```

```
void setPointX(int index, double x)
```

```
double getPointY(int index)
```

```
void setPointY(int index, double y)
```

6. В классе TabulatedFunction описаны методы, изменяющие количество точек табулированной функции:

```
# nano functions/TabulatedFunction.java
```

```
void deletePoint(int index)
```

```
void addPoint(FunctionPoint point)
```

7. Для проверки работы классов создан класс Main:

```
#nano Main.java
```

В методе main создан экземпляр класса TabulatedFunction

```
java Main
```

Точки функции (через конструктор):

```
(0.0, 0.0)
```

```
(1.0, 1.0)
```

```
(2.0, 4.0)
```

```
(3.0, 9.0)
```

Значения функции:

```
f(-1.0) = NaN (вне области)
```

$f(0.5) = 0.5$
 $f(1.0) = 1.0$
 $f(1.5) = 2.5$
 $f(2.0) = 4.0$
 $f(3.0) = 9.0$
 $f(4.0) = \text{NaN}$ (вне области)

Меняем точку с индексом 1 на (1, 2):

Точки после изменения:

(0.0, 0.0)
(1.0, 2.0)
(2.0, 4.0)
(3.0, 9.0)

Пробуем задать $x=2.5$ для индекса 1:

Точка 1: (1.0, 2.0)

Удаляем точку с индексом 1:

Точки после удаления:

(0.0, 0.0)
(2.0, 4.0)
(3.0, 9.0)

Добавляем точку (1.5, 2.25):

Точки после добавления:

(0.0, 0.0)
(1.5, 2.25)
(2.0, 4.0)
(3.0, 9.0)

Задания:

1. Создаю пакет functions

`mkdir functions`

2. Создан класс FunctionPoint.java в пакете functions (в начале класса прописываю `package functions`, тем самым создаю пакет functions).

`nano functions/FunctionPoint.java`

В классе описываю конструкторы:

- `FunctionPoint(double x, double y)` – создаёт объект точки с заданными координатами;
- `FunctionPoint(FunctionPoint point)` – создаёт объект точки с теми же координатами, что у указанной точки;
- `FunctionPoint()` – создаёт точку с координатами (0; 0).

3. Создан класс TabulatedFunction в пакете functions.

`TabulatedFunction(double leftX, double rightX, int pointsCount)`

`nano functions/TabulatedFunction.java`

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

4. Описаны методы:

`double getLeftDomainBorder()`

`double getRightDomainBorder()`

`double getFunctionValue(double x)`

5. В классе TabulatedFunction описаны методы, необходимые для работы с точками табулированной функции:

```
int getPointsCount()
void setPoint(int index, FunctionPoint point)
double getPointX(int index)
void setPointX(int index, double x)
double getPointY(int index)
void setPointY(int index, double y)
```

6. В классе TabulatedFunction описаны методы, изменяющие количество точек табулированной функции:

```
void deletePoint(int index)
void addPoint(FunctionPoint point)
```

7. Для проверки работы классов создан класс Main:
В методе main создан экземпляр класса TabulatedFunction

```
java Main
```

Точки функции (через конструктор):

```
(0.0, 0.0)
(1.0, 1.0)
(2.0, 4.0)
(3.0, 9.0)
```

Значения функции:

```
f(-1.0) = NaN (вне области)
f(0.5) = 0.5
f(1.0) = 1.0
f(1.5) = 2.5
f(2.0) = 4.0
f(3.0) = 9.0
f(4.0) = NaN (вне области)
```

Меняем точку с индексом 1 на (1, 2):

Точки после изменения:

```
(0.0, 0.0)
(1.0, 2.0)
(2.0, 4.0)
(3.0, 9.0)
```

Пробуем задать $x=2.5$ для индекса 1:

Точка 1: (1.0, 2.0)

Удаляем точку с индексом 1:

Точки после удаления:

```
(0.0, 0.0)
(2.0, 4.0)
(3.0, 9.0)
```

Добавляем точку (1.5, 2.25):

Точки после добавления:

```
(0.0, 0.0)
(1.5, 2.25)
(2.0, 4.0)
(3.0, 9.0)
```

