OOP Course - Ex0

In this assignment we were asked to write a class of polynom, and all sorts of important functions to perform on it. Writing the assignment was divided into two classes: Monom and Polynom.

Authors: Meir Nizri and Avihay Bernholtz

Monom

The Monom class represents a simple monom of shape ax^b, where a is a real number and b is an integer (summed a none negative), see: https://en.wikipedia.org/wiki/Monomial. We defined monom so that there can be spaces everywhere (e,g: 2 x ^ 3), as long as the main structure we described is preserved. Writing capital 'X' instead of 'x' will cause an exception. There are some special monoms:

- Real number followed by x - the power of the monom is 1.
- Real number only - the power of the monom is 0.

The class implements function and support simple operations as: construction, value at x, derivative, add and multiply.

Field Summary

Fields		
Modifier and Type	**Field**	**Description**
`static java.util.Comparator<Monom>`	`_Comp`	The Constant _Comp.
`static double`	`EPSILON`	The Constant EPSILON.
`static Monom`	`MINUS1`	The Constant MINUS1.
`static Monom`	`ZERO`	The Constant ZERO.

Constructor Summary

Constructors	
Constructor	**Description**
`Monom(double a, int b)`	Constructs and initializes a monom. a - coefficient, b - power.
`Monom(String s)`	Constructs and initializes a monom from String.
`Monom(Monom ot)`	Constructs a deep copy of the monom.

- ## Method Summary

Modifier and Type	Method	Description
void	**add**(**Monom** m)	Add monom to this monom.
Monom	**derivative**()	this method returns the derivative of the monom.
boolean	**equals**(**Monom** d)	check if equals.
double	**f**(double x)	This method calculate the value of the monom for given x.
double	**get_coefficient**()	Gets the coefficient.
int	**get_power**()	Gets the power.
static java.util.Comparator<**Monom**>	**getComp**()	Gets the comp.
boolean	**isZero**()	Checks if is zero.
void	**multiply**(**Monom** d)	Multiply monom to this monom.
java.lang.String	**toString**()	Print the monom to the screen, in format a_x^_b.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Polynom

This class represents a general Polynom: f(x) = a_1X^b_1 + a_2*X^b_2 +...+ a_n*Xb_n, where: a_1, a_2 ... a_n are real numbers and b_1, b_2 ... b_n are integer (summed a none negative). all the elements in the polynom are standard monoms as we described in the monom class. We defined Polynom so that there can be spaces everywhere (e,g: 2x^ 3 – 5 x), as long as the main structure we described is preserved.
This Polynom can get different monoms with equal powers, The constructor will add all monoms of the same power and order all the elements from highest power to the lowest.

Polynom implements function and support simple operations as: construction, value at x, add, subtract, multiply functionality, it also should support the following:
1. Riemann's Integral: https://en.wikipedia.org/wiki/Riemann_integral
2. Finding a numerical value between two values (currently support root only f(x)=0).
3. Derivative

- ## Constructor Summary

Constructors	
Constructor	**Description**
`Polynom()`	Constructs the zero polynom.
`Polynom(String s)`	init a Polynom from a String such as: {"x", "3+1.4X^3-34x", "(2x^2-4)*(-1.2x-7.1)", "(3-3.4x+1)*((3.1x-1.2)-(3X^2-3.1))"};

- ## Method Summary

Modifier and Type	Method	Description
`void`	`add(Monom m1)`	Add m1 to this Polynom.
`void`	`add (Polynom_able p1 )`	Add p1 to this Polynom.
`double`	`area(double x0, double x1, double eps)`	Compute a Riman's integral from x0 to x1 in eps steps.
`Polynom_able`	`copy()`	create a deep copy of this Polynom.
`Polynom_able`	`derivative()`	Compute a new Polynom which is the derivative of this Polynom.
`boolean`	`equals (Polynom_able p1 )`	Test if this Polynom is logically equals to p1.
`double`	`f(double x)`	calculate the value of the polynom when inserted value of x
`boolean`	`isZero()`	Test if this is the Zero Polynom.
`java.util.Iterator<Monom>`	`iteretor()`	find the iterator of the polynom, which is a pointer to the start of the polynom
`void`	`multiply (Monom m1)`	Multiply this Polynom by Monom m1.
`void`	`multiply (Polynom_able p1 )`	Multiply this Polynom by p1.
`double`	`root(double x0, double x1, double eps)`	Compute a value x' (x0<=x'<=x1) for with |f(x')| < eps assuming (f(x0)*f(x1)<=0, else should throws runtimeException computes f(x') such that: (i) x0<=x'<=x1 && (ii) |f(x')|<eps< div=""> </eps<>

Modifier and Type	Method	Description
void	**sortAndSumEqualDegree**()	Sort the polynom and sum monoms of equal degree.
void	**substract** (**Polynom_able** p1)	Subtract p1 from this Polynom.
java.lang.String	**toString**()	print the polynom

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait