

Netanel Davidov, 208252684

This file is intended to show and explain the purpose of the project, what it does, and what capabilities it gives to those who use it.

file:///C:/Users/orez1/OneDrive/Desktop/Ex0/doc/myMath/Polynom_able.html

PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

PREV CLASS NEXT CLASS FRAMES NO FRAMES ALL CLASSES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

myMath

Interface Polynom_able

All Superinterfaces:
cont_function, function

All Known Implementing Classes:
Polynom

```
public interface Polynom_able
    extends cont_function
```

This interface represents a general Polynomial: $f(x) = a_1x^{b_1} + a_2x^{b_2} \dots a_nx^{b_n}$, where: $a_1, a_2 \dots a_n$ are real numbers and $b_1=0 \dots b_n$ are none negative integers (naturals) For formal definitions see: <https://en.wikipedia.org/wiki/Polynomial> Such polygon has the following functionality: 1. Init: 1.1 Init(String) 1.2 Init() // zero Polygon 1.3 Polynom copy() // deep copy semantics 2. Math: 2.1 void add(Polygon p) // add p to this Polygon 2.2 void subtract(Polygon p) // subtract p from this Polygon 2.3 void multiply(Polygon p) // multiply this Polygon by p 3. Utils 3.1 isZero() 3.2 Polynom derivative() // returns a new Polygon of the derivative ("NIGZERET"). 3.3 double f(x) // return this Polygon value at p(x) 3.4 boolean equals(Polygon p) // returns true iff for any x: this.f(x) == p.f(x) 3.5 double root(double xo, double x1, double eps) // assuming (f(xo)*f(x1)<=0, returns f(x2) such that: // (i) xo<=x2<=x1 & (ii) (f(x2)

Author:
Netanel

Method Summary

All Methods Instance Methods Abstract Methods

Modifier and Type	Method and Description
void	add(Monom m1)

Before us we will see a Polynom_able interface which contains statements on many functions which require implementation in the classes that will be implement the interface. This interface represents a general polynomial of the form :

$f(x) = a_1X^{b_1} + a_2X^{b_2} \dots a_nX^{b_n}$, where: $a_1, a_2 \dots a_n$ are real numbers and $b_1=0 \dots b_n$ are none negative integers (naturals). A general view can be seen as a polynomial composed of a collection of many monom. Each such monom contains two fields: coefficient and power, which distinguish between monom and monom ,therefore, the polynomial itself is defined as a collection.

In this project the user inserts a particular polynomial and gets the option to perform the following operations on this polynomial like : initialization, add polynomial with polynomial, subtract between 2 polynomial, multiply, derivative, equal between to polynoms, copy polynomial, the value of $f(x)$ (of interface function), root approximate value (by numerical method), area by compute Rimman's integral by calculate sum of rectangle in eps size step.

It is also possible to see that polynom_able is also a extends to the cont_function interface, which is an interface for continuous functions. Since polynom is a continuous function, it will inherit this interface and therefore receives an additional function / function that can be performed by Rimman's integral(area).

myMath

Class Polynom

java.lang.Object
myMath.Polynom

All implemented interfaces:
cont_function, function, Polynom_able

```
public class Polynom
extends java.lang.Object
implements Polynom_able
```

This class represents a Polynom with add, 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

Author:
Netanel

Constructor Summary

Constructors	
Constructor and Description	
Polynom()	Class constructor creating a data structure (ArrayList) that will hold objects Monom.
Polynom(java.lang.String polynom)	

The class that implements the Polynom_able interface is the Polynom class. The class definition is to define a collection of monoms that will be the polynomial.

The Polynom can be captured as the creation of a new Polynom object that contains a collection of monom objects, or in the form of a string that the constructor receives, converts it to the expected polynomial (pay attention to input input correctly).

The Monom class mentioned above defines what is a normal monom and also performs actions such as: addition, subtraction, multiplication, derivative and comparison between monom.

These actions are performed on a single monom, so the Polynom class that performs these operations on its polynom, consisting of a collection of monoms, will often use this class because it acts directly and briefly on the polynomial monoms.

Class Monom_Comperator

java.lang.Object
myMath.Monom_Comperator

All Implemented Interfaces:
java.util.Comparator<Monom>

```
public class Monom_Comperator
extends java.lang.Object
implements java.util.Comparator<Monom>
```

this class represente a Comperator

Author:
Netanel

Constructor Summary

Constructors

Constructor and Description
Monom_Comperator()

Method Summary

All Methods **Instance Methods** **Concrete Methods**

Modifier and Type	Method and Description
int	compare(Monom firstM, Monom secondM)

The Monom_Comperator class is a class that defines the object attribute created from this class and is an object whose main function is a comparison function. This class function is used to compare Monom's strengths for various operations that use this class object (such as sorting)