# Project summary

In this project we create an interface representing a function of type y=f(x) that is a combination of monoms, creating one polynom on which we perform mathematical operations using different methods for each sort. The project is divided in multiple interfaces and classes.

## Function interface

This is the superinterface of this project. Subinterfaces: cont\_function, Polynom\_able Implementing Classes: Monom, Polynom

Purpose : Represents a simple function of type y=f(x), where both y and x are real numbers.

Method type : double

Method name : f(double x)

Method parameters : x - variable

Method returns : function of type y=f(x)

## Cont\_function interface

It is one of Function subinterfaces and so the method f(double x) is inherited from it. Subinterfaces: Polynom\_able Implementing Class: Polynom

Purpose: Represents a continuance function.

Method type : double

Method name : area(double x0, double x1, double eps)

Method details : Compute a Riman's integral from x0 to x1 in eps steps.

Method parameters : x0 - starting pooint

x1 - end point

eps - positive step value

Method returns : the approximated area above X-axis below this function bounded in the range of [x0,x1]

## Class Monom

This class creates a **"monom"** of shape ax^b on which we perform mathematical operations such as adding a monom to another one, multiplying two monoms, calculating the value of a monom after x is given a numerical value, or derivating a monom. This class also implements the interface Function and has two private fields; coefficient, that is of type double and power, that is of type integer.

### Constructors

Monom(): this is the default constructor

Monom(double a, int b): this constructor is given two parameters, one of type double that is set as the coefficient of this monom and one of type integer that is set as the power of x.

Monom(Monom ot): this is the copy constructor.

Monom(java.lang.String str): this is the string constructor that helps us print a monom of proper shape a\*x^b.

### Methods

* **Add**

Method type : void

Method details : adds Monom m to this monom only if the powers of the variables are equal.

Method parameters : Monom m

* **Derivative**

Method type : Monom

Method details : operates derivative of this monom.

Method returns : the monom after derivative.

* **f**

Method type : double

Method details : this method is given a numerical value to x and calculates the monom value

Method parameters : x of type double

Method returns : the multiplication of the value with the coefficient (also in consideration of the power)

* **Multiply**

Method type : void

Method details : gets a monom m and multiply it with this monom.

Method parameters : Monom m

* **get\_power**

Method type : int

Method returns : the power of the variable x in this monom.

* **get\_coefficient**

Method type : int

Method returns : the coefficient placed before the variable x in this monom.

* **set\_power**

Method type : void

Method details : sets the numerical value b as power of variable x .

Method parameter : b of type integer

* **set\_coefficient**

Method type : void

Method details : sets the numerical value a as coefficient before variable x.

Method parameter : a of type double

## Class Monom\_Comperator

This class implements java.util.Comparator and use a method to compare two monoms; to verify if they are equals or not.

Method type : int

Method name : compare

Method details : compare betwen two monoms

Method parameters : Monom m1, Monom m2

Method returns : if they are equals it returns 0, if the power of m1 is bigger than the power of m2 it returns -1 and if not it returns 1.

## Polynom\_able interface

This interface represents a general polynom: f(x) = a\_1X^b\_1 + a\_2X^b\_2 ... a\_nXb\_n, where: a\_1, a\_2 ... a\_n are real numbers and b\_1=0 are none negative integers (naturals). Such polygon has the following functionality: polynoms constructors, mathematical operations functions and a String type constructor that are defined in implemented class Polynom.

## Class Polynom

This class represents a general **polynom**: f(x) = a\_1X^b\_1 + a\_2X^b\_2 ... a\_nXb\_n, where: a\_1, a\_2 ... a\_n are real numbers and b\_1=0 are none negative integers (naturals). Such polynon has the following functionality: polynoms constructors, mathematical operations functions and a String type constructor. All implemented interfaces: cont\_function, function, Polynom\_able. This class uses two private fields;one that uses the commpare method Monom\_Comperator and one that the array list method ArrayList for polynom.

### Constructors

Polynom(): this is the default constructor that creates an array list filled with monoms.

Polynom(Polynom p): this is the copy constructor

Polynom(java.lang.String s): this is the string constructor that helps us print a polynom using char characters and symbols according to each function.

### Methods

* **Add(Monom)**

Method type : void

Method details : adds Monom m to this polynom.

Method parameters : Monom m

* **Add(Polynom\_able)**

Method type : void

Method details : adds polynom\_able to this polynom.

Method parameters : Polynom\_able p1

* **Substract**

Method type : void

Method details : substract between this polynom to Polynom\_able p1

Method parameter : Polynom\_able p1

* **Derivative**

Method type : Polynom\_able

Method details : operates derivative of this polynom.

Method returns : a new polynom after derivative of the original polynom.

* **Multiply**

Method type : void

Method details : gets a polynom and multiply it with our polynom.

Method parameters : Polynom\_able p1

* **f**

Method type : double

Method details : this method is given one numerical value to x for the entire polynom and calculates the sum of all the values of the monoms.

Method parameters : x of type double

Method returns : the sum of all the values of the monoms in the polynom (also in consideration of the power).

* **Area**

Method type : double

Method details : Compute a Riman's integral from x0 to x1 in eps steps.

Method parameters : x0 - starting pooint

x1 - end point

eps - positive step value

Method returns : the approximated area above X-axis below this function bounded in the range of [x0,x1]

* **Copy**

Method type : Polynom\_able

Method details : returns a copy of the polynom

* **Equals**

Method type : boolean

Method details : compare between two polynoms if they are equals one to the other.

Method parameter : Polynom\_able p1

Method returns : true if the two polynoms are exactly the same.

* **isZero**

Method type : boolean

Method details : returns true if the polynom is empty

* **Iterator**

Method type : Iterator<Monom>

Method details : this method allows us to search an array list using three functions:

hasNext() : returns true if the iteration has more elements.

next() : returns the next element in the iteration.

void remove() : removes from the underlying collection the last element returned by this iterator.

* **Root**

Method type : double

Method details : The Bisection Method is a successive approximation method that narrows down an interval that contains a root of the function f(x).

This Method is given an initial interval that contains a root and then cuts it into 2 halves and check which half interval contains a root of the function.

It''ll then keep cut the interval in halves until the resulting interval is extremely small, the root is then approximately equal to any value in the final (very small) interval.

Method parameter : double x0, double x1, double eps

Method returns : the bisected root.

* **getPolynom**

Method type : java.util.ArrayList<Monom>

Method details : this method gets a polynom and returns this polynom using array List.

* **areaNew**

Method type : double

Method details : Calculate the area under the x axe of a function in a specific range only

Method parameters : two values for the range and a value for epsilon

Method returns : the area

## Class Graph

## In this class we use a tool from the site [*https://www.commentcamarche.net/*](https://www.commentcamarche.net/)that draws a sketch of the graph of a given function.

### Methods

## ExtremePoints

## Method type : void

## Method details : This method gets a function and a range and find the extreme points of the function

Method parameter : a string, the polynom and two parameters that represent a range

* **Graph­\_print**

## Method type : void

## Method details : This method gets a function and points and draw them in a graph.

Method parameter : a string, an array list and a range

* **Main**

## Test class

In this class we apply every methods of each class, changing the polynoms or monoms and printing them using a main method.

## Authors

Shalhevet Gamliel & Naomi Oyer

Bas du formulaire

Haut du formulaire

.