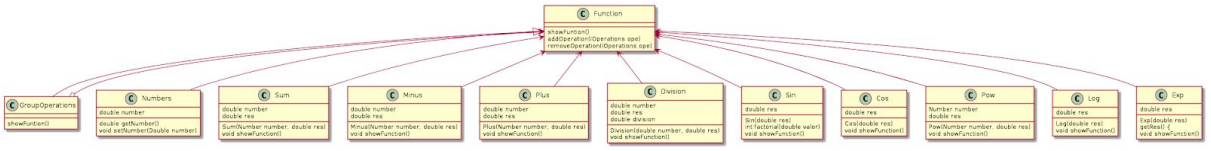
Raül de Arriba 1390196 y Hernan Espinosa 1391108

Automatic differentiation

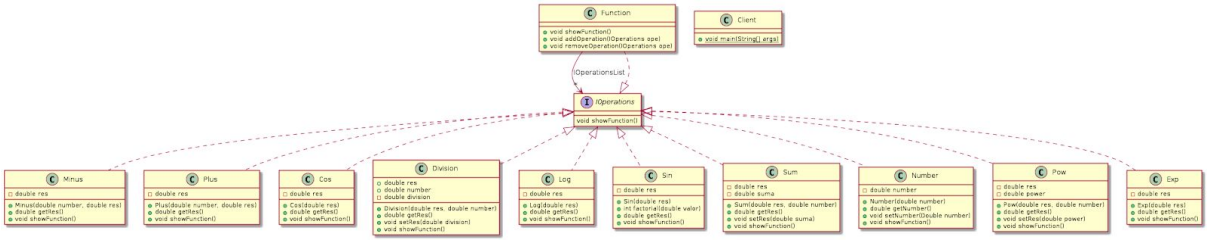
1.- Analyze the problem draw a diagram using plantuml to design a software able to represent any function f (x) that combines numbers, +, −, ×, /, sin, cos, () k , log and exp.

Después de analizar el problema, podemos observar que se puede dividir en pequeñas partes.
El objetivo es computar una función grande y el resultado compararla con la misma función resuelta con la librería java Math.
La función a computar está compuesta por pequeñas funciones o operaciones más simples por ese motivo hemos visto que el problema se resuelve con un patrón composite donde cada hoja es una operación para acabar construyendo la función.
Con este razonamiento hemos hecho el siguiente diagrama:



[imagen adjunta en el documento] El uml está en el fichero diagram.uml dentro de src

Además adjuntamos el diagrama generado automáticamente con el código que está más legible pero vemos que continua siendo un patrón composite



[imagen adjunta en el documento]

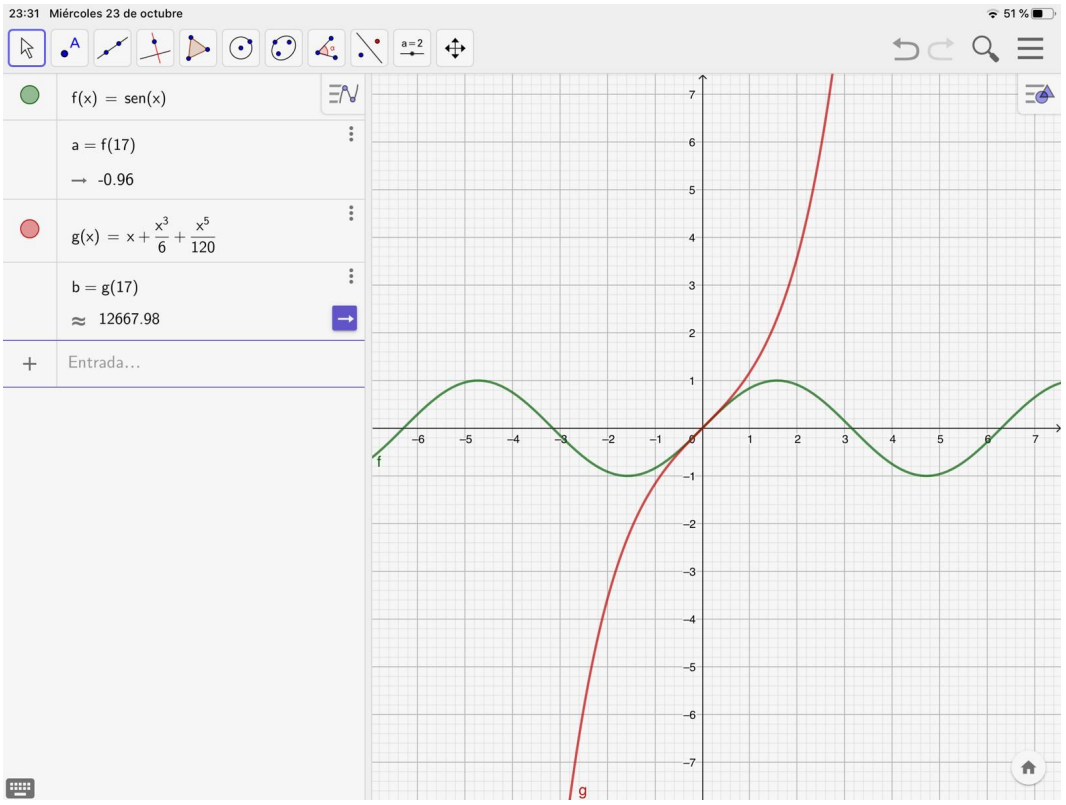
2.- Implement the design in java console to enable the computation of any f (x 0) and f 0 (x 0) for any given x 0 . The expression to evaluate should be hardcoded : Don't implement any expression parser or calculator interface. If we want to change the expression, we will edit the main, change its code, recompile and execute.

Hemos creado todas las clases y prácticamente todas estan programadas, falta el log i la exp. Cabe destacar que nos sale un valor extraño en el resultado ya que para hacer el sin() hemos utilizado el método de aproximación de taylor y no hemos conseguido sacar un valor coherente. El cos() tambien da un valor extraño ya que lo calculamos a partir del seno aislando la ecuación cos^2 + sin^2 = 1.

Raül de Arriba 1390196 y Hernan Espinosa 1391108

3.- Perform some computations with functions like: sin(2+x^2), sin(x)/x, tan(x), showing the real result (using a java math expression), the result computed with your code, the derivatives in both cases, and the diferences in results.

Podemos observar que el valor es muy extraño, el motivo es porque la aproximación de Taylor que hemos aplicado pertenece a una aproximación cercana a 0, adjuntamos imagen con la comparación del seno(verde) con la aproximación de orden 3(rojo). En el programa está hasta orden 10.




```
1  package differenciation;
2
3  public class Client {
4
5      public static void main(String[] args) {
6          System.out.println("Expressió a avaluar : sin(pi/2 + x^2) amb x = 4.0");
7
8          Number x = new Number(4.0);
9          Number pi = new Number(Math.PI);
10         Number number = new Number(2.0);
11         Division div = new Division(pi.getNumber(), number.getNumber());
12         Pow pow = new Pow(x.getNumber(), number.getNumber());
13         Sum suma = new Sum(div.getRes(), pow.getRes());
14         Sin sin = new Sin(suma.getRes());
15
16         Function loss = new Function();
17
18         loss.addOperation(sin);
19         System.out.println("solució computada");
20         loss.showFunction();
21
22         double real = Math.sin(pi.getNumber()/number.getNumber() + Math.pow(x.getNumber(), number.getNumber()));
23         System.out.println("solució real");
24         System.out.println(real);
25
26
27
```

Problems Javadoc Declaration Console PlantUML

```
<terminated> Client [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (23 oct. 2019 21:51:53)
Expressió a avaluar : sin(pi/2 + x^2) amb x = 4.0
solució computada
6342.499670914965
solució real
-0.9576594803233841
Diferencia valor 6343.4573303952875
```

Raül de Arriba 1390196 y Hernan Espinosa 1391108