Université Paris Diderot
Licence Informatique

Programmation Orientée Objet
Année 2009–2010

Devoir n°2

Fonctions de $\mathbb{R}$ dans $\mathbb{R}$

On veut pouvoir représenter en Java une fonction $f : \mathbb{R} \to \mathbb{R}$ comme un objet `f` tel que pour tout réel x, représenté en Java par le `double x`, la valeur $f(x)$ soit obtenue par `f.valeur(x)`

On créera alors une interface ou une classe abstraite (à vous de voir) `Fonction`, de telle sorte que toute fonction soit représentée par une instance d'une classe qui étendra la classe abstraite (ou implémentera l'interface, à vous de voir) `Fonction`.

Exemple. Si on définit la classe suivante :

```
public class Loga /* extends ou implements ? */ Fonction {
  private double a;
  public Loga (double a) {
    this.a = a ;
  }
  public double valeur (double x) {
    if (x <= 0.0) {
      return 0.0 ;
    } else {
      return Math.log(x) / Math.log(this.a) ;
    }
  }
}
```

alors, pour tout réel $a > 0$ représenté par le `double a`, l'objet `new Loga(a)` représente la fonction :

$$\begin{array}{rcl} \mathbb{R} & \to & \mathbb{R} \\ x \leq 0 & \mapsto & 0 \\ x > 0 & \mapsto & \log_a x \end{array}$$

On veut pouvoir également définir la somme, le produit, l'opposée et la composée de deux fonctions quelconques, selon les formules :

- $(f + g)(x) = f(x) + g(x)$
- $(f \times g)(x) = f(x) \times g(x)$
- $(-f)(x) = -(f(x))$
- $(f \circ g)(x) = f(g(x))$

Parmi toutes ces fonctions, on veut pouvoir aussi définir des fonctions C^∞, c'est-à-dire des fonctions indéfiniment dérivables en tout point de $\mathbb{R}$. Pour de telles fonctions, la dérivée existe en tout point de $\mathbb{R}$ et est elle-même C^∞.

Pour ces fonctions, on veut pouvoir calculer la dérivée sous la forme d'une fonction approchée en s'inspirant de la formule :

$$f'(x) = \lim_{\delta \to 0} \frac{f(x+\delta) - f(x)}{\delta}$$

Mais on veut pouvoir également calculer la fonction dérivée sous forme exacte **à la place** de la formule approchée, lorsque l'on connaît la fonction.

Exemple. Dans le cas du cosinus et du sinus hyperboliques :

```
public class CosinusHyperbolique /* extends ou implements ? */ Cinfini {
  public double valeur (double x) {
    return Math.cosh(x);
  }
  public Cinfini derivee () {
    return new SinusHyperbolique();
  }
}

public class SinusHyperbolique /* extends ou implements ? */ Cinfini {
  public double valeur (double x) {
    return Math.sinh(x);
  }
  public Cinfini derivee () {
    return new CosinusHyperbolique();
  }
}
```

On veut pouvoir également calculer les dérivées des sommes, produits, opposées et composées de deux fonctions f et g lorsque ces dernières sont C^∞, suivant les formules :

- $(f+g)' = f' + g'$
- $(f \times g)' = f' \times g + f \times g'$
- $(-f)' = -(f')$
- $(f \circ g)' = (f' \circ g) \times g'$

Question. Modéliser le problème en Java en prenant en compte tous ces paramètres.

Application. Utiliser votre modélisation pour représenter tout en même temps **les** polynômes à coefficients réels, les fonctions cosinus, sinus et arctangente [1] (voir la classe `java.lang.Math`), ainsi que des fonctions C^∞ dont la dérivée première ne s'exprime pas facilement, ainsi que des fonctions non C^∞ (mais néanmoins définies sur tout $\mathbb{R}$).

1. **Indication.** Soit

$$f(x) = \arctan'(x) = \frac{1}{1+x^2}$$

Alors on a

$$f'(x) = -\frac{2x}{(1+x^2)^2} = -2x \times (f(x))^2$$