INFORMATIQUE — Mias 2 — TP 5

Autour de l'héritage

Récupération des fichiers

Les fichiers utilisés dans ce TP sont disponibles dans le répertoire `Info Mias-SM\Java\TP5` sur `jupiter`.

Avant de commencer le travail, il vous faut copier une version personnelle des textes Java. Pour celà, effectuez les opérations suivantes :

1. créez un nouveau répertoire `TP5` dans le répertoire `Java` de votre répertoire personnel (`Z:`);
2. copiez les deux fichiers `.java` du répertoire `TP5` sur `jupiter` dans votre répertoire `TP5` (utilisez les raccourcis `Ctrl-C` et `Ctrl-V`, ou cliquez sur le bouton droit et sélectionnez les libellés `copier` et `coller`);
3. lancez l'application `Emacs` (par le menu `Démarrer`).

Exercice 1 : Jouons un peu

L'application que nous vous proposons est un petit calculateur de séries (type développements limités de fonctions) qui permet de voir comment évolue la série ainsi que la convergence des termes de la suite. Dans la version que nous donnons, la suite calculée (fichier `SuiteRecurrente.java`) est une suite récurrente trés simple : l'identité. Peu utile dans la pratique, elle pourra vous permettre de comprendre le fonctionnement général de l'application.

Compilez les fichiers Java et « jouez » avec l'application pour en comprendre le fonctionnement.

Exercice 2 : Analysons le code

Dans la suite, nous vous demandons de réaliser un ensemble de calculateurs pour des suites particulières. Le principe de cette réalisation sera de définir les suites par le mécanisme d'héritage appliqué à partir de la classe `SuiteRecurrente`.

Avant de se lancer dans la programmation, il est préférable d'analyser le programme en répondant aux questions suivantes :

- Donnez une explication (rapide) du rôle de toutes les méthodes de la classe `SuiteRecurrente`.
- Pourquoi le nom de la suite est-il mis dans une méthode et pas dans une variable d'instance?
- Quelles sont les méthodes de la classe `SuiteRecurrente` à redéfinir pour spécialiser au calcul d'une nouvelle suite?
- Faut-il modifier l'interface? Où?

.../...

Exercice 3 : Héritons

En utilisant l'héritage, écrivez des applications qui permettent de calculer les suites suivantes :

1. exp(x) par la formule $\sum_{i=0}^{i=n} x^i/i!$,
2. sin(x) par son développement limité,
3. cos(x) par son développement limité,
4. $\log_2(x)$ (logarithme en base 2) par son développement limité.

Vous testerez soigneusement chaque nouvelle application.

Exercice 4 : Une application plus générale

Lorsqu'on lance une application Java , il est possible de passer des arguments à la fonction `main` qui est le *point d'entrée* du programme.

Si on tape `java InterfaceSuite exp log ar3`, le tableau qui est le paramétre de la fonction `main` reçoit trois éléments (sous forme de chaînes de caractères) correspondant aux trois derniers « mots » de la commandes. Ainsi :

```
args[0] --> "exp"
args[1] --> "log"
args[2] --> "ar3"
```

Réécrivez la classe `InterfaceSuite` de sorte que :

- il soit possible de mettre les arguments suivants à la commande : `log`, `exp`, `sin`, `cos` (vous pouvez utiliser d'autres chaînes si vous le souhaitez),
- il puisse y avoir autant d'arguments qu'on le souhaite,
- pour chaque argument, un calculateur (une fenêtre) soit créé,
- si un argument n'est pas reconnu (mal orthographié par exemple), il est simplement ignoré et
- s'il n'y a pas d'argument, un calculateur avec la suite identité est créé.

Pour tester la valeur des arguments, vous pourrez utiliser la méthode `equals`, en écrivant par exemple `arg[i].equals("log")`.