


E304A : Laboratoire de concepts informatiques

Travail 2 : Super calculatrice

Lors de la quatrième séance de laboratoire, vous avez eu le loisir de faire vos premiers pas avec l'introspection de code et la réflexivité. Ce deuxième travail va mobiliser ces compétences pour la réalisation d'une super calculatrice. L'idée du travail est de proposer une calculatrice avec une interface en ligne de commande, qui permet de réaliser des opérations plus ou moins avancées. Par exemple, une exécution pourrait ressembler à :

```
=== Bienvenue dans Calculator 3.1 ===
>>> cos 0
1
>>> derive x^2+2x-3
2x+2
>>> max 2 6 1 4
6
>>> derive sin(x)
UnsupportedFunctionError: Only polynomials are accepted.
```

Le format des commandes est simple : on commence avec le nom de la commande suivi d'une liste de paramètres séparés par des espaces. Dans l'exemple ci-dessus, les commandes `cos` et `derive` reçoivent un paramètre et la commande `max` en reçoit un nombre arbitraire. Une commande peut être malformée ou son exécution peut provoquer une erreur, dans lesquels cas un message d'affiche.

Définissez d'abord une classe abstraite générique `Command` qui représente une commande, et qui possèdera certainement une méthode abstraite `execute`. Ensuite, définissez une sous-classe par commande que vous voulez proposer. Vous ne devez pas hardcoder les différentes commandes disponibles, mais elles seront chargées, au démarrage de votre programme, par réflexion. 

1 Contenu du travail

Vous allez devoir franchir plusieurs étapes dans le cadre de ce travail :

1. Établir votre plan de bataille, à savoir définir les différentes classes dont vous allez avoir besoin pour les différents aspects de l'application (gérer l'interface en ligne de commande, parser les commandes, etc.).
2. Implémentez votre solution avec une seule commande de test facile.
3. Ajouter le chargement dynamique des commandes par réflexion depuis un dossier spécifié et le tester avec la commande de test facile.
4. Ajouter des nouvelles commandes en écrivant de nouvelles classes.

2 Échéance et livrable

En plus du code commenté à remettre, vous réaliser un court rapport (quelques pages maximum) décrivant notamment les commandes que vous avez décidées de réaliser. Votre rapport contiendra également un bref mode d'emploi de votre programme. Le rapport est à remettre au **format PDF**.

Le travail est à faire pour le **vendredi 27 novembre 2016 à 18h30**. Vous devez rendre votre rapport, ainsi qu'une archive ZIP contenant votre code, sur l'espace Claco dédié, avant l'échéance, sachant qu'**aucun retard** ne sera toléré.

Le travail est à réaliser en binômes.

3 Grille d'évaluation

Le projet sera évalué sur un total de **20 points** dont voici la répartition :

- | | |
|---|--------------------|
| 1. L'implémentation | (14 points) |
| — La programmation orientée objet est utilisée de manière adéquate | (4 points) |
| — Bonne utilisation du concept de classe abstraite (1 point) | |
| — Bonne utilisation du concept d'héritage (1 point) | |
| — Bonne utilisation des objets avec des propriétés et méthodes (1 point) | |
| — Les relations entres classes sont adéquates (1 point) | |
| — La réflexion a été utilisée de manière correct | (3,5 points) |
| — Au moins trois commandes suffisamment différentes, et avec un nombre et un type différent de paramètres, ont été définies | (4,5 points) |
| — Les erreurs de commandes sont bien gérées | (1 points) |
| — Les commentaires dans le code sont suffisants et adéquats | (1 point) |
| 2. L'application | (3 points) |
| — L'interface en mode console est présente et facile à utiliser | (2 points) |
| — L'application fonctionne correctement | (1 point) |
| 3. Le rapport | (3 points) |
| — Le rapport est remis dans le bon format | (1 point) |
| — Le rapport est concis et apporte des informations utiles et pertinentes | (1 point) |
| — Le mode d'emploi est précis, concis et complet | (1 point) |