UFR
Sciences et
Techniques

Licence 2

25 Rue Philippe Lebon
B.P. 540
76058 Le Havre Cedex

Examen de Spécialisation Informatique

Vendredi 6 Mai 2011
Durée : 3 heures
Documents, téléphones et calculatrices non autorisés.
Il est possible d'utiliser les réponses à une question non traitée pour résoudre les autres questions.

I *Exceptions*

On souhaite calculer la factorielle d'un nombre entier positif et utiliser le traitement des exceptions dans le programme.

$$\forall n \in \mathbb{N}^* \; n! = n * (n-1)! \text{ et } 0! = 1$$

I.1 Définissez une classe de base `Factorielle` en complétant le squelette ci-après. On considère dans cette question que l'utilisateur fournit un entier "correct".

```
public class Factorielle {
  public static void main(String[] args) {
    int input = Integer.parseInt(args[0]);
    long resultat = factorielle(input);
    System.out.println(input + "!=" + resultat);
  }

  public static long factorielle(int x) {
  // ...
  }
}
```

I.2 On considère maintenant que l'utilisateur est susceptible de fournir une information incorrecte qui provoquera alors une exception de type `NumberFormatException`, modifier votre code afin d'informer l'utilisateur à l'aide d'un message à la console que son entrée est incorrecte. Vous afficherez également l'état de la pile d'exécution.

I.3 En plus des erreurs de saisie, on considère également que l'utilisateur peut rentrer un nombre négatif. Dans ce cas votre méthode `factorielle()` devra lever une erreur `PasDef` que vous aurez définie.

- Définir la classe `PasDef` ;
- Modifier la méthode `factorielle()` pour qu'elle génère cette exception lorsque x est négatif.
- Modifier si nécessaire d'autres parties de votre programme.

I.4 Cette même exception doit être levée quand la factorielle demandée sera trop grande, modifiez en conséquence la méthode `factorielle()`.

II *Polymorphisme*

Dans cet exercice on cherche à décrire les données d'un jeu simulant des combats de magiciens. Il existe trois types de cartes : les terrains, les créatures et les sortilèges.

1. Les terrains possèdent une couleur (parmi 5 : blanc('B'), bleu ('b'), noir ('n'), rouge ('r') et vert ('v')). ;
2. Les créatures possèdent un nom, un nombre de points de nuisance et un nombre de points de vie.
3. Les sortilèges possèdent un nom et une explication sous forme de texte.

De plus, chaque carte, indépendamment de son type, possède un coût. Celui d'un terrain est 0.

```
public class Magic {
 public static void main(String[] args) {
```

```
 Jeu maMain = new Jeu(10);

 maMain.piocher(new Terrain('b'));
 maMain.piocher(new Creature(6, "Golem", 4, 6));
 maMain.piocher(new Sortilege(1, "Croissance Gigantesque",
                "La écrature écible gagne +3/+3 jusquà' la fin du tour"));

 System.out.println("J'ai en stock :");
 maMain.afficher();
 maMain.joue();
 }
}
```

II.1 Définissez les classes nécessaires permettant de représenter des cartes de différents types. Chaque classe aura un constructeur permettant de spécifier la/les valeurs de ses attributs. De plus, chaque constructeur devra afficher le type de la carte.

II.2 Ajoutez ensuite aux cartes une méthode `afficher()` qui, pour toute carte, affiche son coût et la valeur de ses arguments spécifiques. Modifier les classes nécessaires.

II.3 Créez une classe `Jeu` pour représenter un jeu de cartes, c'est-à-dire un tableau de telles cartes. Cette classe devra avoir une méthode `piocher()` permettant d'ajouter une carte au jeu. On supposera qu'un jeu comporte au plus 10 cartes. Le jeu comportera également une méthode `joue()` permettant de jouer une carte. Pour simplifier, on jouera les cartes dans l'ordre où elles sont stockées dans le jeu, et on mettra la carte jouée à `null` dans le jeu de cartes.

II.4 Expliquez où et comment vous avez utilisé le polymorphisme dans cet exercice.

III *Fichier*

Écrire un programme Java qui prend en entrée sur la ligne de commande le nom d'un fichier contenant le code de classe Java et supprime tous les commentaires (`//Ceci est un commentaire`) le résultat est stocké dans un autre fichier dont le nom est également fourni en ligne de commande.