TP, quatrième série d'exercices

Benoît Valiron
benoit.valiron@lipn.univ-paris13.fr

6 décembre 2010

Exercice 1.

reprenez la classe `Point`.

1. Implémentez une fonction `public void loadScreen()` qui lit les coordonnées d'un point entré au clavier, une coordonnée par ligne. Ajoutez du texte pour rendre l'interface plus conviviale (Par exemple : ``` ``entrez la premiere coordonnée...'' ``` puis ``` ``entrez la deuxieme...'' ```)
2. Modifiez la classe `TestPoint` : Créez un point `p`, et modifiez ses coordonnées avec la méthode juste créée. Affichez les coordonnées du point à l'écran pour vous assurez que ça marche.
3. Essayez votre programme en plaçant des lettres dans les coordonnées : quelle exception est levée ? Allez voir dans la documentation et donnez sa place dans la hiérarchie des exceptions :

   `/usr/share/doc/java-sdk-docs-1.6.0/html/api/index.html`

   Est-on forcé de gèrer de type d'exception ?
4. Modifiez votre programme pour attraper l'exception et avertir l'utilisateur du programme.

Exercice 2.

1. Créez deux méthodes `void writeText(String name)` et `void readText(String name)` qui écrivent et lisent dans un fichier `name` les valeurs d'un point **en clair**, et modifie le point courant en conséquence.

   Pour l'ecriture : le format du point tt x,y.

   Pour la lecture : utilisez la méthode `split` de la classe `String` afin de séparer les coordonnées.
2. Dans la classe `TestPoint`, créez un point, demandez à l'utilisateur le nom d'un fichier avec des coordonnées en texte, lisez-le et placez dans le point les coordonnées lues. Imprimez le point à l'écran.

   Prenez soin de gèrer les exceptions : Si le nom du fichier est invalide, redemandez à l'utilisateur, ou si la lecture du fichier est fautive, demandez au clavier les coordonnées qui manquent.

Exercice 3.

Dans cet exercice, on reprend la classe `TamponEntiers`.

1. Créez une nouvelle exception `TamponInvalide`.
2. Implémentez un constructeur qui prend en argument un objet `Reader`, supposé contenir des entiers, chacun sur une ligne de texte. Le constructeur lit le flux et place les entiers dans un objet `TamponEntiers`, jusqu'à ce que le flux soit vide. Si une erreur survient (par exemple on ne lit pas un entier), la méthode lèvera une exception `TamponEntiers`.
3. Utilisez votre constructeur pour lire au clavier des entiers (on indique la fin d'un fichier avec control-D). Imprimez le `TamponEntiers` ainsi créé.
4. Dans `TamponEntiers`, faites une méthode d'écriture en clair dans un fichier et une méthode de lecture d'un fichier en clair (un entier par ligne).
5. (Bonus) Créez une classe `ConcatEntier` avec une méthode `main`. Le programme ainsi créé est supposé prendre pour ses deux premiers arguments des noms de fichiers avec des flux d'entiers. La méthode lit les fichiers, crée deux objets `TamponEntiers`, les concatène dans un troisième `TamponEntier` et affiche le résultat.

   On gèrera les exceptions levées et on affichera des messages d'erreur propres en cas de problème.

Exercice 4.

1. Reprenez la question 1 de l'exercice 2, mais avec des méthodes `void writeBin(String name)` et `void readBin(String name)`, qui écrivent et lisent en binaire (et non pas en texte).
2. Commentez le code déjà écrit dans `TestPoint`, et continuez à la suite.

   Dans la classe `TestPoint`, créez un point `p` de valeur `(1.2, 3.4)` et un point `q = (0,0)`. Écrivez le point `p` dans un fichier `test.bin`. Lisez le fichier `test.bin` et placez le résultat dans `q`. Affichez le point `q` à l'écran, pour vous assurer du succès de l'opération.
3. Quel est la taille du fichier `test.bin` ? Pourquoi ? Ouvrez-le avec un éditeur de texte. Que contient-il ?