

Java TP Héritage - Exceptions

Vous allez devoir écrire un programme gérant les principales formes géométriques, et doté d'un mécanisme permettant de créer ces formes à partir d'un fichier texte.

Ce fichier pourra avoir l'aspect suivant :

CERCLE

10 15 12 // coordonnées point + rayon

QUADRILATERE

1 1 10 10 20 20 50 50 // coordonnées 4 points

TRIANGLE

1 1 10 10 20 20 // coordonnées 3 points

Sa structure est donc la suivante :

- Une ligne stockant l'intitulé en clair
- Une ligne stockant les paramètres, séparés par des points

1) Ecrire les classes de base permettant de gérer ces formes géométriques, en utilisant une structure d'héritage (pour l'instant, la visualisation des formes sera un simple affichage texte).

2) Créez une classe qui contiendra un outil : une méthode convertissant une chaîne de caractères de type "10 15 12" en tableau d'entiers. On utilisera pour cela la classe StringTokenizer, ou encore la méthode split() de String (consultez la JavaDoc pour plus d'infos)

3) Créez la classe Surface, qui jouera le rôle de container pour les différentes formes. Cette surface sera limitée par une taille rectangulaire (largeur, longueur).

3) Ecrivez le mécanisme capable de générer les instances de formes géométriques à partir d'un fichier texte.

Utilisez pour cela au maximum les principes de généricité (et le polymorphisme) : par exemple, toute forme prendra en paramètre du constructeur la 2^e ligne du fichier, et décortiquera ensuite cette ligne pour en déduire ses attributs.

Le nom du fichier sera passé en paramètre par le biais du tableau String[] args de la méthode main)

Lire un fichier s'effectue de la manière suivante :

```
String ligne;  
FileReader fr=new FileReader("fichier");  
BufferedReader br=new BufferedReader(fr);  
while((ligne=br.readLine())!=null)  
{  
    // traitement de la ligne  
}  
br.close();  
fr.close();
```

N'oubliez pas de gérer les exceptions liées à ces lignes de code !

4) Gérez une nouvelle exception, *MalformedGeometricException*, qui sera levée lorsque l'on tentera de construire une forme avec des paramètres incorrects (par exemple : "10 12" pour un cercle).

5) Gérez une exception spéciale, *OutOfAreaException*, qui sera levée lors de la tentative d'utilisation de points avec des coordonnées en dehors de l'aire.

- 6) Gérez également l'affichage (en mode texte, via un toString()) de toutes les formes géométriques, là aussi via polymorphisme.
- 7) Créez une interface nommée Colorable, qui impliquera l'implémentation d'une méthode setColor(String couleur) dans chacune des formes. Créez une méthode dans Surface permettant ainsi de changer la couleur de toutes les formes.
- 8) Avec la méthode sort de Collections, et l'interface Comparable, créez une méthode dans Surface qui permettra de trier les formes stockées par taille.
- 9) Utilisez l'interface Serializable pour proposer une fonction de lecture / écriture dans un fichier des formes stockées dans la surface.