Programmation Orientée Objets

TD n°8 : Exceptions

Exercice 1 : Une première exception

Modifier la classe `EnsembleForme` pour qu'une tentative d'accès à une forme dont le numéro n'existe pas dans le tableau des formes lève une exception « `IndexOutOfBoundsException` ».
Intercepter l'exception dans le `main` de `Test.java` et afficher un message d'erreur.

Exercice 2 : Une exception personnalisée

On considère maintenant qu'un `Polygone` ayant un ou plusieurs sommets situés en (0,0) est incomplet. Le calcul du périmètre lève alors une exception personnalisée de type `SommetNul`.
Quelles modifications apporter, à quelle(s) classe(s) ?
Intercepter l'exception dans le `main` et afficher un message d'erreur.

Exercice 3 : Message

Modifier le traitement de l'exception de l'exercice 2 afin de transmettre le numéro du sommet qui a déclenché l'interception. Afficher ce numéro dans le message d'erreur.

Exercice 4 : Propagation

Dans la méthode `sommePerimetres()` de la classe `EnsembleForme`, intercepter l'exception `SommetNul`. Propager cette exception sous la forme d'une exception de type `PerimNul`. Intercepter cette exception dans le `main` et afficher un message d'erreur.

Exercice 5 : Chaînage

On souhaite maintenant aussi considérer comme incomplets les cercles ayant leur centre en (0,0) **ou** un rayon égal à zéro.
Créer un type d'exceptions `PointNul` dans l'interface `Forme`.
Créer deux types d'exceptions `CentreNul` et `RayonNul`, qui héritent de `PointNul`, dans la classe `Cercle`.
Remarque : l'exception `SommetNul` doit maintenant elle aussi hériter de `PointNul`.
Mettre en place un chaînage d'exceptions de manière à afficher la cause de l'exception `PerimNul` dans le `main` à l'aide de la méthode `printStackTrace()`, qui met en évidence le chaînage d'exceptions et permet de savoir si l'exception initialement levée était une exception `RayonNul` ou `SommetNul`.