TD : GESTION DES EXCEPTIONS

EXERCICE 1 : Contrôle et capture des exceptions

Soit les deux classes d'exception suivantes :

```
class Exc1 extends RuntimeException { .... }
class Exc2 extends IOException { .... }
```

Soit la classe suivante :

```
public class TestException {
      public void f1(){throw new Exc1(); }
      public void f2(){throw new Exc2(); }
      .....
}
```

1) Expliquer pourquoi f1() se compile sans problème, tandis qu'une erreur de compilation se produit lors de l'analyse de f2() ?
2) Proposer une solution pour résoudre ce problème de compilation.

Exercice 2 : Exceptions personnalisées avec passage de paramètre

Réaliser une classe **EntNat** permettant de gérer des entiers naturels (positifs ou nuls) et disposant :

- d'un constructeur avec un argument de type int ; il générera une exception de type *ErrConst* si la valeur de son argument est négative ;
- un accesseur en lecture getN() qui fournira sous forme d'un int la valeur encapsulée dans un objet de type EntNat ;
- un accesseur en écriture setN() qui modifiera la valeur de l'entier naturel grâce à un int passé en paramètre ; cette méthode générera une exception de type *ErrModif* si la valeur passée en paramètre est négative ;
- une méthode décremente() qui décrémente de 1 un objet EntNat ; cette méthode devra pouvoir lever une exception de type *ErrModif* ;
- une méthode de classe – statique donc – décremente(EntNat e) qui décrémente de 1 l'objet passé en paramètre *(c'es juste pour que vous travaillez sur les méthodes de classe, il serait en effet normal d'en faire une méthode d'instance …)*

Écrire une méthode **main** qui utilise les méthodes de la classe EntNat, en capturant les exceptions susceptibles d'être générées.

Exercice 3 : Exceptions personnalisées

Le programme fourni ne fait aucune vérification sur l'intégrité des données servant à la création d'un objet de type `Project`.

Modifiez-le de sorte à :

- ce que ne soient créés que des projets dont le nom et le sujet n'excède pas les 10 caractères ;
- garantir que la durée du projet soit bien un entier positif.

Le nom, sujet et durée seront redemandés à l'utilisateur tant qu'ils sont introduits de façon incorrecte.

Vous introduirez pour cela deux classes d'exceptions personnalisées `WrongDurationException` et `NameTooLongException`

Une `String`, `strNumber`, correspondant à un entier peut être transformée en `int` par l'appel à la méthode statique `parseInt` de la classe `Integer`(`Integer.parseInt(strNumber)`.

Si `strNumber` ne corresponds pas à un `int` une `RunTimeException` de type `NumberFormatException` sera lancée.

Le programme sera proprement modularisé.