R2.03 - Qualité de développement
TP 01 - Première partie - Les exeptions

Axel Lecoeuche, Rémi Synave et Franck Vandewiele
(et merci Sam pour le sujet)

Pour ce TP, vous allez devoir créer des projets java sous Visual Studio Code pour passer des arguments à la méthode `main`. Avant de commencer, assurez vous de savoir comment faire ! (`https://youtu.be/JUH8GzVw9Ik`)

Mécanisme des exceptions

Le mécanisme des exceptions permet une gestion fine des cas imprévus afin de limiter les cas d'erreurs lors de l'exécution d'un programme. C'est un mécanisme qui est utilisé dans de nombreux langages (Delphi, Java, Ada, Python, C++, C#, ...).

Ces langages utilisent un **Système de Gestion d'Exception** qui produit une interruption de l'exécution du programme lorsqu'une exception se produit. Un bloc de code peut alors être exécuté afin de **traiter** l'exception.

Pour certaines applications, il faut éviter l'arrêt pur et simple du programme dans ces circonstances imprévisibles $\Longrightarrow$ traiter les **exceptions**. En Java, les exceptions et leur mécanisme de gestion sont représentés par une hiérarchie d'objets. Le principe :

- Une méthode détecte un cas qu'elle ne sait pas traiter ;
- Elle crée un objet d'une sous-classe de `Exception` qui va contenir les informations décrivant ce qui s'est passé ;
- Le cours normal du programme est interrompu et l'exception est transmise au **gestionnaire d'exception** ;
- Celui-ci va "remonter" les appels de méthodes ayant conduits à l'exécution de la méthode erronée, en cherchant un code de gestion de l'exception écrit par le programmeur ;
- Par défaut, la gestion interne conduit à :
  - afficher les différents appels ayant conduits à l'exception ;
  - stopper le programme.
- le programmeur est censé prévoir une conduite à tenir lorsque l'exception se produit (problème complexe ...).

Exercice 1 : observer les exceptions sur un programme simple

Compilez et exécutez le programme ci-dessous.

```
public class Carre {

  public static void main(String args[]){
    int i = Integer.parseInt(args[0]);
    int carre = i*i;

    System.out.println("i x i = " + carre);
  }
}// Carre
```

Trouvez les entrées qui provoquent des exceptions. Quelle est la classe de ces exceptions ? (Trouvez en deux !)

Capture des exceptions

Par défaut, le gestionnaire d'exception stoppe le programme et affiche le chemin d'exécution ayant mené à l'exception.

Le programmeur peut tenter de gérer le cas d'erreur en **capturant** l'exception et en traitant "proprement" le problème.

Principe :

1. Identifier un bloc d'instructions susceptible de générer une erreur ;
2. **Essayer** de l'exécuter ;
3. **Capturer** l'exception si elle se produit.

— **Essai** d'exécution : clause `try`.
— **Capture** d'une exception : clause `catch`.

— Syntaxe :

```
try {
  // bloc d'instructions
} catch(TypeException1 e1){
  // ce qu'il faut faire dans ce cas
} catch(TypeException2 e2){
  // ce qu'il faut faire dans ce cas
}
```

Exercice 2 : Gestion d'exceptions

Modifiez le programme précédent pour que son comportement soit de ce type :

```
java Carre 5
i x i = 25

java Carre
ERREUR : pas de parametre

java Carre xyz
ERREUR : parametre incorrect xyz
```

Exercice 3 : Utilisation des exceptions pour relancer une demande

Attraper une exception permet également au programmeur de prévoir le cas où l'utilisateur fait une faute de frappe ou une erreur dans les informations demandées. Il est alors possible d'indiquer à l'utilisateur une erreur et de relancer la demande.

En partant du code se trouvant dans le fichier `MainFichier.java`, faites en sorte d'attraper l'exception qui peut être levé et d'indiquer un message d'erreur comme :
`Le fichier test.txt est introuvable.`
Suite à ce message d'erreur, la demande du nom du fichier doit être renouvelé jusqu'à ce qu'un fichier soit lu et affiché. Vous aurait besoin d'ajouter un boucle et sûrement un booleén.

Transmettre une exception

Il est également possible de ne pas capturer une exception à l'intérieur d'une méthode mais de simplement la propager à la méthode qui l'a appelée. Pour cela, on indique, par le mot clé `throws` dans la déclaration de la méthode que l'exception sera propagée :

```
void maMethode() throws Exception1, Exception2 {

// ici, il n'est pas necessaire de capture Exception1 et Exception2

}
```