TD8 : Exceptions

Objectifs

- Exceptions
- Lancement (throw), capture (try-catch) et propagation (throws) d'exceptions prédéfinies
- Lancement, capture et propagation d'exceptions créées par le programmeur
- Passage de paramètres sur la ligne de commande

Exercices

Rappel : Les exceptions sont un mécanisme de gestion des erreurs. Il existe 3 catégories d'exceptions : les exceptions qui étendent la classe Exception qui doivent obligatoirement être gérées par le programme, les exceptions qui étendent la classe RunTimeException qui peuvent être gérées par le programme, et les erreurs critiques qui étendent la classe Error qui ne sont pas censées être gérées en temps normal.
Rappel : Toute instance de la classe Exception doit obligatoirement être capturée ou bien signalée comme étant propagée par toute méthode susceptible de la lever.

- Pour capturer une exception :

```
try {
    instructions qui peuvent lever une exception
} catch (MonException me) {
    System.out.println(me.toString());
} catch (AutreException ae) {
    System.out.println(ae.getMessage());
} finally {
    instructions toujours executees
}
```

- Pour signaler une erreur, on va lever / lancer une exception, pour cela il faut créer un nouvel objet : `throw new MonException();`
- Pour définir un nouveau type d'exception, il faut écrire une classe qui hérite de la classe Exception : `public class MonException extends Exception {...}`
- Pour déléguer / transmettre / propager une exception pour qu'elle soit capturée par une autre méthode : `public void maMethode () throws MonException {...}`

Exercice 65 – Capture dans le main d'une exception prédéfinie (try catch)

Q 65.1 Soit classe TestAttrapePas0 ci-dessous. Que se passe-t-il lors de l'exécution ?

```
public class TestAttrapePas0{
    public static void main(String[] args){
        int [] tab= {1,2,3,4,5};
        for (int i=0; i<15; i++)
            System.out.print(tab[i] + ".");
    }
}
```

Q 65.2 La méthode getMessage() de l'exception ArrayIndexOutOfBoundsException retourne la position dans le tableau à laquelle l'erreur s'est produite. Modifier la classe TestAttrapePas0 pour capturer cette exception et afficher le texte : "Exception : depassement des bornes a la position 5" quand l'exception se produit.

Exercice 66 – Try, catch, throw, throws, création d'une exception utilisateur

Q 66.1 Écrire une classe TestAttrapePas1 dans laquelle on définira une méthode de classe moyenne(String[] tab) qui, étant donné un tableau de chaînes de caractères représentant des notes (entiers entre 0 et 20), rend la moyenne entière de ces notes. Testez cette méthode dans un main, en affichant la moyenne des notes passées en argument sur la ligne de commande, sans capturer l'exception éventuellement levée.

Indications :

- Utiliser la méthode Integer.parseInt qui transforrme une chaîne de caractères en entier et lève une exception NumberFormatException si la chaˆıne n'est pas un entier.
- Les arguments qui sont passés en ligne de commande sont récupérables par le tableau String[] args passé en paramètre de la méthode main.

Q 66.2 Que donnent les exécutions suivantes :

1. javaTestAttrapePas1 10 12 16 18
2. javaTestAttrapePas1 12 1j 10 13 15
3. javaTestAttrapePas1

Q 66.3 Dans une classe TestAttrape2, reécrire une méthode moyenne(String[] tab)qui calcule la moyenne des notes de tab, mais capture cette fois l'exception levée si une note n'est pas entière et la traite en affichant le message la note n'est pas entière .

1. Où peut-on attraper l'exception NumberFormatException ?
2. Que se passe-t-il si aucune des notes n'est pas entière ou s'il n'y a aucune note ?

Q 66.4 Écrire une classe AucuneNoteEntiereException dérivée de la classe Exception. Dans une classe TestAttrape3, réécrire la méthode moyenne qui lancera une instance de la classe AucuneNoteEntiereException lorsque ce cas se présentera. Cette exception sera capturée dans le main.

Q 66.5 Que donne l'exécution de la commande javaTestAttrape3 mm reg 6r c5 mm ?

Q 66.6 Créer de même une classe PasEntre0et20Exception qui servira à traiter les cas où une note serait négative ou strictement supérieure à 20. Où faut-il capturer cette nouvelle exception ? Modifier le programme dans une classe TestIntervalle pour qu'il lève et capture aussi cette exception. Que donne l'exécution de la commande javaTestIntervalle -10 -3 45 -78 -6 21 ?

Exercice 67 – EntierBorne (throw,throws)

Le but de l'exercice est de définir une classe EntierBorne qui représente tous les entiers entre -10 000 et +10 000 et se prémunisse des dépassements de ces bornes. On testera au fur et à mesure les méthodes écrites. Note : toutes les exceptions seront capturées dans le main.

Q 67.1 Écrire dans une classe TestEntierBorne la méthode main qui saisit une valeur entière. On utilisera obligatoirement la méthode saisirLigne de la classe Clavier non standard qui affiche un message et lit un String, puis la méthode parseInt de la classe Integer (voir la documentation en ligne pour cette méthode) pour transformer la chaîne saisie en entier. Dans le cas où la saisie n'est pas un entier, cette méthode peut lever l'exception NumberFormatException.
Que se passe-t-il à l'exécution si la saisie n'est pas entière ? Expliquez.

Q 67.2 Traiter maintenant l'exception levée dans le main. Ajouter les instructions pour que le main s'endorme pendant *n* secondes en utilisant la méthode sleepde la classe Thread qui lève une exception de type InterruptedException.

Q 67.3 Écrire la classe EntierBorne qui est une classe enveloppe du type simple int, i.e. qui "enveloppe" une variable d'instance de type int dans un objet de cette classe. Écrire le constructeur à un paramètre de type intqui peut lever l'exception HorsBornesException si la valeur qui est passée en paramètre est plus grande que 10000 ou plus petite que −10000, et la méthode toString(). On définira pour cela la classe HorsBornesException.

Q 67.4 Définir la méthode EntierBorne somme(EntierBorne i) qui rend un objet EntierBorne dont la valeur est la somme de cet élément et du paramètre. Elle pourra lever sans la capturer l'exception HorsBornesException si la somme est trop grande.

Q 67.5 Définir la méthode EntierBorne divPar(EntierBorne i) qui rend un objet EntierBorne dont la valeur est la division entière de cet élément par le paramètre i. Elle pourra lever l'exception HorsBornesException ou l'exception DivisionParZeroException.

Q 67.6 On définira ensuite la méthode EntierBorne factorielle() qui calcule la factorielle de cet élément. Elle pourra, en plus de l'exception HorsBornesException, lever l'exception IllegalArgumentException dans le cas où *n* serait négatif.

Q 67.7 Créer un jeu de tests pour ce programme, en réfléchissant aux différents cas possibles et les tester dans le main.