Calculatrice

# TP du Module 7 – Les exceptions

|  |
| --- |
| Avant de démarrer ce TP, il convient d’avoir suivi les vidéos des modules 1 à 7. Son objectif est de comprendre plus en profondeur le mécanisme des exceptions et de savoir créer des exceptions personnalisées. |

|  |
| --- |
| **Durée estimée** |
| 2 heures |

Énoncé

L’objectif est de créer une mini calculatrice réalisant des opérations d’addition, de soustraction, de multiplication, de division entière et de reste de celle-ci.

Cette calculatrice manipule des entiers de type int. Avec le type int, les valeurs doivent être comprises entre -2 147 483 648 et 2 147 483 647. Si une variable de type int est initialisée à 2 000 000 000 et que la variable est multipliée par deux, la variable contiendra -294 967 296 ! Il y a eu un « dépassement de capacité ». Cela ne provoque malheureusement ni erreur ni levée d’exception.

Un autre problème avec les opérations sur les entiers est que la division par zéro n’est pas mathématiquement définie. Lorsqu’une telle opération est réalisée, une exception de type ArithmeticException est levée.

La calculatrice devra prendre en compte cela pour éviter d’afficher un résultat faux.

Indications

1 – Création du projet :

* Garder les options par défaut, notamment la séparation des fichiers sources et des fichiers compilés

2 – Création d’une classe DepassementCapaciteException :

Cette classe utilisée pour représenter une exception contrôlée. Le message associé est « Le résultat dépasse la capacité de la calculatrice ».

3 – Création de la classe utilitaire Operation :

Cette classe contient trois méthodes de classe ajouter, soustraire et multiplier (il n’est pas nécessaire de créer une méthode diviser car la division entière ne peut pas provoquer de dépassement de capacité). Elles prennent toutes deux entiers en paramètre et retournant un entier. Elles lèvent toutes les trois des exceptions de type DepassementCapaciteException dans le cas où le calcul provoque un dépassement de la capacité d’un entier de type int.

**Indication :** Une manière simple de tester si un calcul avec des types int dépasse sa capacité est de réaliser le calcul avec des types long et vérifier si les résultats sont différents.

4 – Création de la calculatrice :

Voici un exemple d’exécution attendue :

Saisir un nombre entier :

bonjour !

Saisir un nombre entier :

Saisie incorrecte. Réessayez...

9999999999

Saisir un nombre entier :

La valeur saisie dépasse les capacités de cette calculatrice. Réessayez...

1234567

1234567

opérateur ? (+ - \* / % ou q pour quitter)

\*

Saisir un nombre entier :

987654

1234567

opérateur ? (+ - \* / % ou q pour quitter)

Le résultat dépasse la capacité de la calculatrice

\*

Saisir un nombre entier :

42

1234567 \* 42 = 51851814

opérateur ? (+ - \* / % ou q pour quitter)

/

Saisir un nombre entier :

17

51851814 / 17 = 3050106

opérateur ? (+ - \* / % ou q pour quitter)

/

Saisir un nombre entier :

0

La division par zéro n'est pas définie !

3050106

opérateur ? (+ - \* / % ou q pour quitter)

q

Le programme doit gérer les exceptions provoquées par la saisie erronée pour un nombre entier, le dépassement de capacité lors d’une opération ou de la saisie d’une valeur et la division d’une valeur par zéro.

Solution

Une solution est proposée pour ce TP sous la forme d'un PDF commenté avec le code associé. Ces éléments sont disponibles dans les ressources à télécharger.