

## Calculatrice

---

1. Créez une fonction appelée `calculatrice` qui prend trois arguments : `nombre1` (un nombre), `nombre2` (un nombre), et `operation` (une chaîne de caractères représentant l'opération à effectuer, par exemple : "+", "-", "\*", "/").
2. À l'intérieur de la fonction, utilisez une instruction `try` pour gérer les exceptions qui pourraient se produire lors de l'opération.
3. Dans le bloc `try`, effectuez l'opération demandée (addition, soustraction, multiplication ou division) en fonction de la valeur de `operation`. Assurez-vous de retourner le résultat de l'opération.
4. Si une exception se produit, capturez-la et retournez un message d'erreur approprié. Par exemple, si une division par zéro se produit, retournez "Division par zéro impossible."
5. En dehors du bloc `try`, appelez la fonction `calculatrice` avec différentes valeurs pour `nombre1`, `nombre2` et `operation` pour tester la gestion des exceptions.
6. Affichez le résultat ou le message d'erreur retourné par la fonction.

Voici un exemple d'utilisation de la fonction `calculatrice` :

```
resultat = calculatrice(5, 2, "+") # Résultat attendu : 7
print(resultat)

resultat = calculatrice(10, 0, "/") # Résultat attendu : "Division par zéro impossible."
print(resultat)
```

Assurez-vous de gérer correctement les exceptions et de retourner les messages d'erreur appropriés en cas d'erreur. Bonne chance !

## Exceptions personnalisées

---

L'objectif de cet exercice est de créer une exception personnalisée appelée `ValeurNegativeException` qui sera levée chaque fois qu'une valeur négative est rencontrée. Voici les étapes à suivre :

1. Créez une classe `ValeurNegativeException` qui hérite de la classe de base `Exception`.
2. Définissez un constructeur pour cette classe qui prend un message en argument et appelle le constructeur de la classe de base avec ce message.
3. Dans le constructeur, assurez-vous que le message contient une description claire de l'erreur, par exemple : "La valeur ne peut pas être négative."
4. Créez une fonction appelée `verifier_valeur` qui prend un nombre en argument.

5. À l'intérieur de cette fonction, vérifiez si le nombre est négatif. Si c'est le cas, levez une instance de `ValeurNegativeException` en passant le message approprié.
6. Si le nombre est positif ou nul, affichez un message indiquant que la valeur est valide.
7. En dehors de la fonction `verifier_valeur`, appelez-la avec différentes valeurs pour tester la levée de l'exception personnalisée.
8. Capturez l'exception levée et affichez le message d'erreur.

Voici un exemple d'utilisation de l'exception personnalisée `ValeurNegativeException` :

```
try:
    verifier_valeur(10) # Valeur valide
    verifier_valeur(-5) # Cette ligne devrait lever une exception
except ValeurNegativeException as e:
    print(e) # Affiche "La valeur ne peut pas être négative."
```

Assurez-vous que votre exception personnalisée fonctionne correctement et qu'elle affiche le message d'erreur approprié lorsqu'une valeur négative est rencontrée. Bonne pratique de la gestion d'exceptions personnalisées !