Série de Travaux Pratiques N° 6

Exercice 1 : Division par Zéro

Écrivez une fonction `safe_division(a, b)` qui prend deux arguments et renvoie le résultat de la division de `a` par `b`. Si `b` est zéro, la fonction doit lever une exception `ZeroDivisionError` avec un message approprié.

Exercice 2 : Vérification de Type

Créez une fonction `convert_to_int(value)` qui tente de convertir `value` en entier. Si `value` n'est pas convertible, la fonction doit lever une exception `ValueError` avec un message indiquant que la conversion a échoué.

Exercice 3 : Lecture de Fichier

Écrivez une fonction `read_file(filename)` qui tente d'ouvrir et de lire un fichier. Si le fichier n'existe pas, la fonction doit gérer l'exception `FileNotFoundError` et imprimer un message d'erreur. Utilisez également un bloc `finally` pour garantir que le fichier est fermé.

Exercice 4 : Exceptions Personnalisées

Créez une exception personnalisée `NegativeAgeError`. Écrivez une fonction `set_age(age)` qui lève cette exception si l'âge est négatif. Testez la fonction dans un bloc `try` et gérez l'exception en imprimant un message approprié.

Exercice 5 : Multi-Exceptions

Écrivez une fonction `process_input(user_input)` qui tente de convertir `user_input` en entier et d'effectuer une division par 10. Gérez les exceptions `ValueError` et `ZeroDivisionError` dans des blocs `except` séparés, en affichant des messages d'erreur appropriés.

Exercice 6 : Utilisation de `else` et `finally`

Modifiez l'exercice 1 pour inclure un bloc `else` qui imprime un message indiquant que la division a été effectuée avec succès, et un bloc `finally` qui indique que la fonction est terminée, quel que soit le résultat.

Exercice 7 : Journalisation des Exceptions

Utilisez le module `logging` pour créer une fonction `log_error(message)` qui enregistre les erreurs dans un fichier `error.log`. Modifiez l'exercice 3 pour utiliser cette fonction pour enregistrer une erreur si le fichier n'est pas trouvé.

Exercice 8 : Tests Unitaires pour les Exceptions

Écrivez des tests unitaires pour vérifier que les exceptions sont levées correctement dans les fonctions que vous avez créées dans les exercices précédents. Utilisez le module `unittest` pour cela.

Exercice 9 : Gestion des Exceptions dans les Boucles

Créez une fonction `get_positive_integer()` qui demande à l'utilisateur de saisir un entier positif. Utilisez une boucle pour continuer à demander jusqu'à ce qu'une saisie valide soit fournie. Gérez les exceptions de conversion et vérifiez que l'entier est positif.

Exercice 10 : Combinez Tout

Écrivez un programme qui demande à l'utilisateur de saisir un fichier, puis un entier. Utilisez les concepts de gestion des exceptions pour garantir que le fichier est lu avec succès et que l'entier est valide. Gérer toutes les exceptions appropriées et afficher des messages utiles.