**Упражнение 1: Написать функцию с обработкой ошибок для преобразования строки в число**

Задание: Создать функцию **convert\_to\_int(value),** которая пытается преобразовать строку в целое число. Обработать случаи, когда преобразование невозможно, и когда возникает любое другое исключение.

**Пошаговая инструкция:**

* Создайте функцию **convert\_to\_int(value).**
* Используйте блок **try** для попытки преобразования строки в целое число с помощью функции **int().**
* Обработайте исключение **ValueError**, выведя сообщение о невозможности преобразования.
* Обработайте любое другое исключение, выведя сообщение о неожиданной ошибке.
* Добавьте блок **finally**, который выведет сообщение о завершении попытки преобразования.

**Проверка:**

* Протестируйте функцию с корректной строкой, например, **"123".**
* Протестируйте функцию с некорректной строкой, например, **"abc".**
* Протестируйте функцию с другим типом данных, например, списком **[1, 2, 3].**

**Упражнение 2: Написать функцию с валидацией входных данных и генерацией исключений**

Задание: Создать функцию **validate\_user\_input(data)**, которая принимает словарь с данными пользователя. Функция должна проверять наличие и корректность ключей **name** (строка) и **age** (положительное число). В случае некорректных данных генерировать соответствующие исключения.

**Пошаговая инструкция:**

* Создайте функцию **validate\_user\_input(data).**
* Проверьте, что **data** является словарем. Если нет, вызовите **TypeError**.
* Проверьте, что ключ name присутствует в словаре и его значение является строкой. Если нет, вызовите **ValueError**.
* Проверьте, что ключ **age** присутствует в словаре и его значение является положительным числом. Если нет, вызовите **ValueError**.
* Используйте ключевое слово **from**, чтобы создать цепочку исключений, если это необходимо.

**Проверка:**

* Протестируйте функцию с корректными данными, например, **{"name": "Alice", "age": 30}.**
* Протестируйте функцию с отсутствующим ключом **name**.
* Протестируйте функцию с некорректным значением для **age** (например, строкой или отрицательным числом).
* Протестируйте функцию с некорректным типом входных данных (**например, строкой вместо словаря).**

**Упражнение 3: Написать класс собственного исключения и использовать его в примере кода:**

Задание: Создать класс **NegativeNumberError**, который будет генерироваться, если переданное число отрицательное. Использовать это исключение в функции **check\_positive\_number**.

**Пошаговая инструкция:**

* Создайте класс **NegativeNumberError**, наследующий от **Exception**.
* Переопределите конструктор, чтобы принимать значение числа и сообщение об ошибке.
* Переопределите метод **\_\_str\_\_,** чтобы возвращать информативное сообщение.
* Создайте функцию **check\_positive\_number**, которая проверяет, является ли число положительным. Если число отрицательное, функция должна вызывать исключение **NegativeNumberError**.

**Проверка:**

* Протестируйте функцию с отрицательным числом.
* Протестируйте функцию с положительным числом.
* Убедитесь, что исключение **NegativeNumberError** правильно обрабатывается и выводит информативное сообщение.

**Каждое задание в отдельных файлах и со всеми этапами проверки (GitHub)**