Lesson Summary

This lesson covers additional Python simple statements beyond expressions, assignments, and imports. Specifically it introduces exceptions, raise statements to trigger exceptions, and assert statements to validate conditions.

Key points:

- Exceptions will stop execution of a program when an error occurs
- Raise statements manually raise exception errors
- Assert statements validate conditions, raising errors if they fail
- These mechanisms help handle errors and validate assumptions in code

Reflection Questions

1. What are some common exceptions you might encounter when writing Python code? ↳ Zero Division Error
2. When might you want to manually raise an exception in your code?
   ↳ When we have defined a function but have not defined it behavior yet.
3. How could you use assert statements to validate the inputs to a function?
   ↳ assert condition message
4. What exception handling code could you add to make your programs more robust?
   ↳ Nope, it showed which is our mistaken in this programe
5. Why is it useful to raise errors and handle exceptions in programming?
   ↳ defined which is type of error

Challenges

1. Add a raise statement to throw a custom exception when invalid parameters are passed to a function
2. Use try/except blocks to catch errors and handle them gracefully
3. Validate numeric inputs to functions with assert statements
4. Research built-in Python exceptions and choose ones relevant for a program you are writing
5. Handle possible exceptions from importing external libraries or modules

Code Examples

```python
import pandas as pd

def analyze_series(s):
    """Analyze pandas Series.

    Demonstrates assertions and exception handling.
    """

    # Validate inputs
    assert isinstance(s, pd.Series), "Input must be a pandas Series"

    try:
        # Attempt processing
        mean = s.mean()
        median = s.median()

    # Catch errors
    except Exception as e:
        # Manually raise exception
        raise ValueError("Error analyzing series") from e

    # Assert reasonable results
    assert (mean > 0) & (median > 0), "Mean and median are invalid!"

    return mean, median

s = pd.Series([1, 2, 3])
analyze_series(s)
```

```
(2.0, 2.0)
```