Exception Handling 🌪️ (1.5 hours)

Errors happen! To make sure your programs are **error-proof** and user-friendly, Python provides **Exception Handling**. It's the art of catching errors and handling them gracefully.

- **Basic Structure of `try-except` Blocks** ⚙️
  - `try`: Runs code that might throw an error.
  - `except`: Catches the error, allowing you to respond without crashing.
  - **Example**

```
try:
    with open("nonexistent.txt", "r") as file:
        data = file.read()
except FileNotFoundError:
    print("File not found. Please check the filename.")
```

Advanced Error Handling with `finally` and Custom Errors 🎩

- **`finally`**: Runs no matter what, often used to **clean up** (like closing a file).
- **Custom Errors**: Create custom exceptions for special cases (e.g., `EmptyFileError`).

Example with `finally`:

```
try:
    file = open("sample.txt", "r")
    data = file.read()
except FileNotFoundError:
    print("File not found.")
finally:
    file.close()
```

Best Practices 📝

- **Use `with` for file handling**: Auto-close files, preventing potential leaks.
- **Check file existence** before reading/writing, to avoid crashes.
- **Handle specific exceptions** over general ones (e.g., `FileNotFoundError` instead of `Exception`).
- **Document error messages** clearly for easier debugging and user support.