Exception Handling in Python — an essential concept for writing **safe**, **robust**, and **error-resistant** programs.

What is an Exception?

An **exception** is an error that **occurs during program execution** and disrupts the normal flow of the program.

Example:

```
print(10 / 0)  # ❌ ZeroDivisionError
```

Why Use Exception Handling?

To **catch and handle errors** gracefully instead of crashing your program.

Basic Syntax:

```
try:
    # Code that might raise an error
except SomeError:
    # What to do if that error occurs
```

1. Try–Except Example

```
try:
    x = 10 / 0
except ZeroDivisionError:
    print("Cannot divide by zero.")
```

Output:

Cannot divide by zero.

2. Multiple Except Blocks

```
try:
    a = int("abc")
except ValueError:
    print("Invalid value")
except ZeroDivisionError:
    print("Division by zero")
```

3. Generic Exception

```
try:
    x = int("hello")
except Exception as e:
    print("Error occurred:", e)
```

Output:

```
Error occurred: invalid literal for int() with base 10: 'hello'
```

4. Else Block

Runs **only if no exception** occurs.

```
try:
    print("No errors here!")
except:
    print("Error occurred")
else:
    print("Try block successful")
```

5. Finally Block

Runs **no matter what** — used for cleanup tasks like closing files.

```
try:
    print("Inside try")
finally:
    print("Always runs, error or not")
```

Raising Exceptions Manually

Use `raise` to throw an exception yourself.

```
age = -5
if age < 0:
    raise ValueError("Age cannot be negative")
```

Summary Table

Keyword	Description
`try`	Code to test for errors
`except`	Code to handle the error
`else`	Code to run if no error occurs
`finally`	Code that runs no matter what (cleanup, close)
`raise`	Used to throw an exception manually

📝 Mini Assignment – Exception Handling

Create a file `exception_assignment.py` and solve the following:

Q1. Safe Division

Ask two numbers and divide them. Handle:

- Division by zero
- Non-numeric input

Q2. File Reader with Try-Except

Ask the user for a filename and try to read it.

- Show proper message if file does not exist.

Q3. Raise Custom Exception

Write a function `check_age(age)` that raises an error if age is less than 0.

Q4. Use Finally Block

Use `try-except-finally` to print:

`"All done." (even if error occurs)`