

## → Day 9 : Exception Handling & Debugging

### → Types of Errors in Python

- What is an error?

→ An error is a problem that stops a program from running correctly.  
Errors occur when Python does not understand or cannot execute your code.

- Main types of errors.

Error Type	Meaning.
Syntax Error	Wrong structure / typo, violating writing rules.
Runtime Error	Error while program is running.
Logical Error	Code runs but gives wrong output.

- Syntax Error example.

```
print("Hello
```

This happens because the string is not closed.

- Runtime Error Example

```
print(10/0)
```

This gives ZeroDivisionError because division by zero is not allowed.

- Logical Error Example.

```
print(5+5*2) # Expected 20, but output is 15
```

The logic is wrong, still code runs. (Here multiplication operator has more priority, that's why multiplication occurs first, then addition)

### → What is an Exception?

An exception is a special type of runtime error that can be caught and handled so the program does not crash.

- Try and except.

try allows you to test code that might cause an error, and except allows you to handle error safely.

• syntax :

try :

# code that may cause error

except :

# what to do if that above code gives error.

• example :

try :

x = int(input("Enter a number :"))

print(10/x)

except :

print("Invalid input or division by zero")

# This prevents program from crashing when the user enters bad input.

→ else block

else only runs if no exception occurs in the try block

Note:

It is similar to else block from if-else part, but here it works as else if any exception does not occur skipping except block.

→ Example :

```
try :
    x = int(input("Enter a number: "))
    print(10/x)
except:
    print("Error occurred")
else:
    print("Calculated Successfully")
```

→ finally block.

finally runs no matter what, whether an error occurs or not.

→ Example :

```
try:
    x = int(input("Enter a number: "))
    print(10/x)
except:
    print("Error")
finally:
    print("program ended")
```

This is used for cleanup like closing files or database connections after execution.

## → Multiple Exceptions.

Different type of errors can be handled separately.

→ Example:

try:

```
x = int(input("Enter a number: "))
```

```
print(10/x)
```

except ZeroDivisionError:

```
print("You cannot divide by zero")
```

except ValueError:

```
print("Please enter a valid number")
```

## → Python Exceptions and their meaning.

Exception	Meaning
SyntaxError	Code grammar is wrong
IndentationError	Incorrect spacing or indentation
NameError	Variable or function is not defined.
TypeError	Operation used on wrong data type.
ValueError	Correct type but invalid value.
ZeroDivisionError	Dividing by zero
IndexError	List index is out of range.
KeyError	Dictionary key not found
ImportError	Module cannot be imported.
ModuleNotFoundError	Module does not exist.
FileNotFoundError	File not found.
RecursionError	Function calls itself too many times.