EXCEPTION HANDLING :-

When a Python program meets an error, it stops the execution of the rest of the program. An error in Python might be either an error in the syntax of an expression or a Python exception.

What is an Exception?

An exception in Python is an incident that happens while executing a program that causes the regular course of the program's commands to be disrupted. When a Python code comes across a cond^n it can't handle, it raises an exception. An object in Python that describes an error is called an Exception.

⇒ Try and Except Statement :

Try and except statements are used to catch and handle exceptions in Python. Statements that can raise exceptions are kept inside the try clause and the statements that handle the exception are written inside except clause.

Ex: Here we are trying to access the array element whose index is out of bound and handle the corresponding exception.

```python
a = [1, 2, 3]
try:
    print("2 element :", a[1])

    print(" 4 element :", a[3])

except:
    print("An error occured")
```

Output:

```
2 element : 2

An error occured
```

→ A try statement can have more than one except clause, to specify handlers for different exceptions. Please note that at most one handler will be executed.

```python
try:
    # statement (s)
except Indexerror:
    # statement (s)
except Value Error:
    # statement (s)
```

⇒ **Try with Else Clause :**

In Python, you can also use the else clause on the try-except block which must be present after all the except clauses. The code enters the else block only if the try clause does not raise an exception. ex:

```
def Abc (a,b):
    try:
        c = ((a+b)|(a-b))
    except ZeroDivision Error:
        print ("a/b result in 0")
    else:
        print (c)
Abc (2.0, 3.0)
Abc (3.0, 3.0)
```

Output :

-5.0

a/b result in 0

⇒ **Finally Keyword in Python :**

Python provides a keyword finally, which is always executed after the try and except blocks. The final block always executes after the normal termination of the

try block or after the try block terminates due to some exception.

```
try:
    # Some code

except:
    # Optional block
    # Handling of exception (if required)

else:
    # execute if no exception

finally:
    # Some code .... (always executed)
```