**Python Advance Assignment-7**

**1. What is the try statement for?**

Ans.

Under **try** block we write the codes in which there is a chances of exception. Whatever the codes is, we write those inside the **try** block

**2. What are the two common variations of the try statement?**

Ans.

1) try with except block: When try fails except will execute. When try passess, except do not execute

2) try with except and else: When try will execute then only else block will exceute. If try fails then only except block will exceute not else block

**3. What is the raise statement for?**

Ans.

**raise** keyword is use for raising out the error by ourself. Means we can define the which kind of error we want to raise. We can also simply write down the text message to show the user if error encountered

x **=** "shivansh"

**if** **not** type(x) **is** int:

**raise** ValueError("Only integers are allowed")

**---------------------------------------------------------------------------**

**ValueError** Traceback (most recent call last)

**<ipython-input-6-4db8afca92cc>** in <module>

2

3 **if** **not** type**(**x**)** **is** int**:**

**----> 4 raise** ValueError**("Only integers are allowed")**

**ValueError**: Only integers are allowed

**4. What is the assert statement designed to do, and what other statement is it like?**

Ans.

**assert** statement is for raising assertion error. When any script is not True under the assert keyword, it will raise the assertion error. Else it will simply print.

Another statement which work the same is **raise** keyword

Examples are :

*#raise keyword*

a**=**10

**if** a**==**10:

**raise** Exception('Wrong')

**---------------------------------------------------------------------------**

**Exception** Traceback (most recent call last)

**<ipython-input-10-1f1cddd1229f>** in <module>

4

5 **if** a**==10:**

**----> 6 raise** Exception**('Wrong')**

**Exception**: Wrong

In [12]:

*#assert keyword*

a**=**10

**assert** a**==**1,'Wrong number'

print('right number')

**---------------------------------------------------------------------------**

**AssertionError** Traceback (most recent call last)

**<ipython-input-12-b0a32476773a>** in <module>

3 a**=10**

4

**----> 5 assert** a**==1,'Wrong number'**

6 print**('right number')**

**AssertionError**: Wrong number

**5. What is the with/as statement designed to do, and what other statement is it like?**

Ans.

using **with/as** statement makes the code much compact and readable. While using with staement we do not need to call the file.close() function to close the execution of the resource. The with statement itself ensures proper acquisition and release of resources.

In **try** and **except** block if any file handling operation done like reading or writing and there is no **with/as** statement is used and if there is any error occur then it may happen that it prevents the closing of the file due to which several bugs comes into that. So to avoid this we use **with/as** statement, it helps in ensuring that the resource is properly released and code using the resource propery till it get executed completly.

Example: try: with open('xyz.txt','w') as file:

file.write('this is a file name xyz')

execpt Exception as e:

print(e)

**Similar statement**: It s like try:

file = open('xyz.txt','w')

file.write('This is a xyz file')

except Exception as :

print(e)

else:

print(file.close())

Here if file.close() is required to complete teh effect of the file opration in the fie 'xyz.txt'. Secondaly if ther eis any execption then it causes difficulties in file closing and effecting the file operation insid ethe 'xyz.txt' file