Assertions

What are assertions?
assertion is condition given within program.
This condition can be enabled or disabled during execution of program.
Assertions are used for debugging purpose.

How to create assertion?
Assertion is created using a keyword "assert".

Syntax-1: assert <condition>
Syntax-2: assert <condition> ,message

Example:

```
a=int(input("enter any integer "))
assert a>0,"input number is not greater then zero"
print("continue...")
```

Output

```
enter any integer 5
continue...

enter any integer 0
Traceback (most recent call last):
  File "E:/python5pmjun/extest14.py", line 2, in <module>
    assert a>0,"input number is not greater then zero"
AssertionError: input number is not greater then zero
```

Example:

```
enter first integer 5
enter second integer 2
the division of 5/2=2.5
```

Example:

```
a=int(input("enter first integer "))
b=int(input("enter second integer "))
assert b!=0,"ZeroDivisionError"
c=a/b
print(f'the division of {a}/{b}={c}')
```

Output

```
enter first integer 6
enter second integer 0
Traceback (most recent call last):
  File "E:/python5pmjun/extest15.py", line 3, in <module>
    assert b!=0,"ZeroDivisionError"
AssertionError: ZeroDivisionError
```

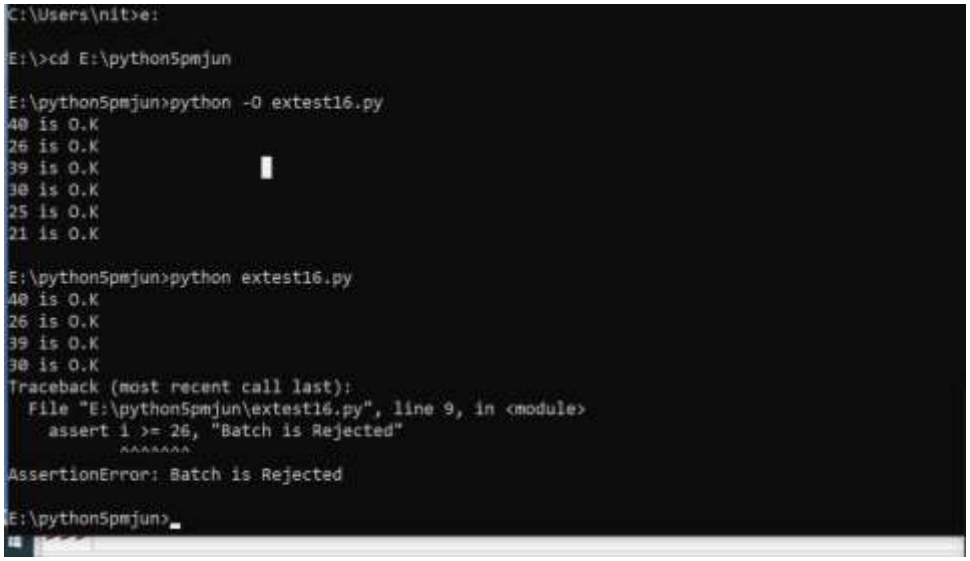
Example:

```
# initializing list of foods temperatures
batch = [ 40, 26, 39, 30, 25, 21]

# initializing cut temperature
cut = 26

# using assert to check for temperature greater than cut
for i in batch:
    assert i >= 26, "Batch is Rejected"
    print (str(i) + " is O.K" )
```

Output

```
40 is O.K
26 is O.K
39 is O.K
30 is O.K
Traceback (most recent call last):
  File "E:/python5pmjun/extest16.py", line 9, in <module>
    assert i >= 26, "Batch is Rejected"
AssertionError: Batch is Rejected
```

Assertions can be disabled by using –O, while executing program at command prompt.



Files or File Handling
==========

What is file?
File is named memory location on secondary storage device (disk).
File is a collection of information or data.
Files are used to save data permanently.

Types of files
There are two types of files

1. Text file
2. Binary file

Text file: In text file data is stored in text format (OR) text file allows only text or string type of data.
Example: .txt, .csv, .json, .rtf,….

Binary file: binary file is collection of bytes (OR) binary file allows only bytes data.
Example: images, audio, video, pdf, doc,…

Basic steps to work with files

1. **Open File**
2. **Read/Write**
3. **Close File**

How to open the file?

open() function

This function opens the given file or path and returns file object.

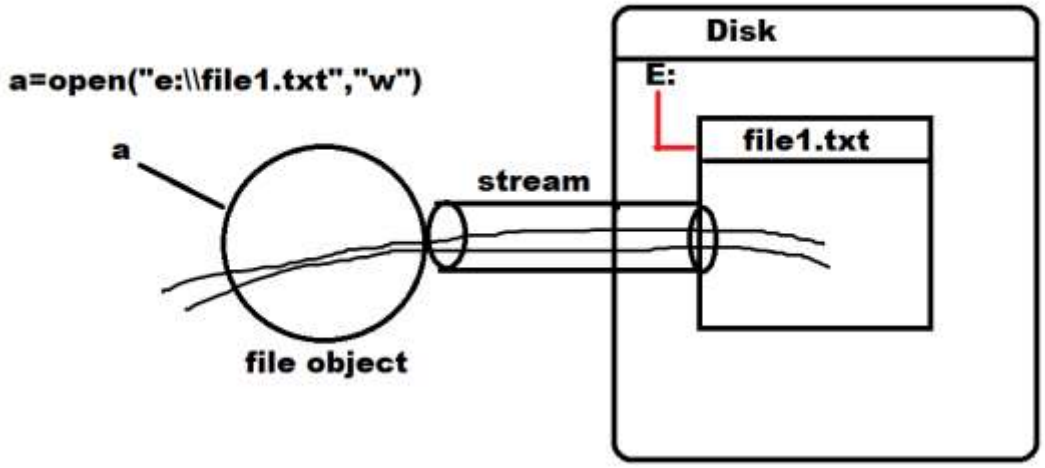
Syntax:
<variable-name>=open("file-name","mode")

file-name is a string, the file-name is given along with given path
file-opening-mode is a string, which define in which file has to opened

Mode	**Description**
w	Writing Create new file by truncating existing file
r	Reading (default) If file exists open for reading else it raises FileNotFoundError
a	Append If file exists add more information If file not exists, create new file and write data
x	Exclusive creating (Writing) If file exists, it raises error If file not exists, it create new file for writing

w+r	This allows writing and reading
r+w	This allows reading and writing (Update)
b	Binary file
t	Text file (default)

Example : wt,rt,at,w,r,a (default type is text)
wb,rb,ab,



Text files
Text file allows only string data.

The following methods are used for reading and writing data into text file

1. write(str)
2. print()
3. read()
4. readline()

write(s, /)
Write the string s to the stream and return the number of characters written.

Example:

```
try:
    fobj=open("e:\\file1.txt","w")
    fobj.write("Jython")
    fobj.write("3.12")
    fobj.write("65")
    fobj.write(str(65))
    fobj.write(str(1.5))
except TypeError:
    print("invalid type or type must be string")
finally:
    fobj.close()
```

Output

Output is saved in file.txt file created in E: drive

Example:

```
# Write a program to input student information into student.txt file

import sys
try:
    f=open("e:\\student.txt","w")
    while True:
        rollno=int(input("Rollno: "))
        name=input("Name: ")
        sub1=int(input("Subject1 :"))
        sub2=int(input("Subject2 :"))
        print(rollno,name,sub1,sub2,file=f)
        ans=input("Add another student ?")
        if ans=="no":
            break
except:
    print(sys.exc_info())
finally:
    f.close()
```

Output

```
Rollno: 1
```

Name: naresh
Subject1 :60
Subject2 :70
Add another student ?yes
Rollno: 2
Name: suresh
Subject1 :80
Subject2 :90
Add another student ?yes
Rollno: 3
Name: kishore
Subject1 :40
Subject2 :30
Add another student ?yes
Rollno: 4
Name: ramesh
Subject1 :60
Subject2 :34
Add another student ?yes
Rollno: 5
Name: kiran
Subject1 :70
Subject2 :80
Add another student ?no

read(*size=-1*, /)
Read and return at most *size* characters from the stream as a single str. If *size* is negative or None, reads until EOF.

Example:

```
# Write a program to read content of file1.txt file

f=open("e:\\file1.txt","r")
ch1=f.read(1)
print(ch1)
ch2=f.read(1)
```

```
print(ch2)
s=f.read()
print(s)
```

Output

```
J
y
thon3.1265651.5
```

Example:

```
# Write a program to count vowels inside file1.txt

f=open("e:\\file1.txt","r")
c=0
while True:
    ch=f.read(1)
    if ch=='':
        break
    if ch in "aeiou":
        c=c+1

print(f'Vowel Count {c}')
```

Output

```
Vowel Count 9
```

Example:

```
# Write a program to copy content of one file inside another file

f1=open("e:\\file1.txt","r")
f2=open("e:\\file2.txt","w")
s=f1.read()
f2.write(s)
f1.close()
f2.close()
print("file copied...")
```

Output
file copied...

readline(*size=-1*, /)
Read until newline or EOF and return a single str. If the stream is already at EOF, an empty string is returned.
If *size* is specified, at most *size* characters will be read.