Date 11/12/21

CHAPTER → 8

Library Functions

इस chapter में math, random & string तीन module हैं।

Introduction :-

पायथन में कुछ pre-define module पहले से होते हैं. जिन्हे library module कहा जाता हैं। प्रत्येक module में कुछ built in फंक्शन शामिल होते हैं। जिन्हे library function कहा जाता हैं।

प्रत्येक फंक्शन किसी विशेष कार्य को करने के लिए होता हैं।

Ex:- (Math, Random and string) → Module

Ex:- abs(), sqrt(), pow(), str(), islower() → Function

किसी भी module के फंक्शन का प्रयोग करने के लिए सर्वप्रथम import function के द्वारा वह module प्रोग्राम में add किया जाता हैं।

Module तीन प्रकार से import किया जा सकता हैं –

1. To import entire module
2. Import selected object of module
3. Import all object of module

(1.) **To import entire module** :– Import Statement में फंक्शन के नाम का प्रयोग करके किसी module को import किया जा सकता हैं।

Syntax: import <module name>, <mn> .....
Example: import math, string
Ex.: print(math.sqrt(25))
Output 5

(2.) **Import Selected object of module :-**

From import Statement के द्वारा किसी module के Selected function को import किया जा सकता है।

Syntax: From <module name> import <function name>

Ex:
```
From math import pi, sqrt, pow
print (pi())      → 3.141
print (sqrt(25))  → 5
print (pow(3,2))  → 9
print (math.sqrt(36)) → Error
```

(3.) **Import all object of module :-**

From import Statement का प्रयोग करके किसी module के सभी object को भी import किया जा सकता है।

Module के सभी फंक्शन import करने के लिए import के बाद '*' का प्रयोग किया जाता है।

Syntax: From <module name> import *

Example:
```
From math import *
```

Math Module :- Math module को import करने के बाद इसके किसी भी फंक्शन का प्रयोग किया जा सकता है। Math module में विभिन्न प्रकार के फंक्शन मौजूद है।

(1) **abs( )** → यह फंक्शन निरपेक्ष मान return करता है।

from import math

print (abs (-32.8))

→ 32.8

import math

print (math.abs (-32.8))

→ 32.8

(2) sqrt ( ) → वर्ग मूल निकालने के लिए।

Syntax: math.sqrt (<any number>)

Example: print (math.sqrt (121))

Output → 11

(3) exp ( ) → यह फंक्शन दी गई संख्या के (e^x) को Calculate करके exponent Value return करता है।

e = 2.718281

Syntax: math.exp (<any number>)

Ex. print (math.exp (3))

20.08554

(4) ceil ( ) → इस फंक्शन का प्रयोग दिये गये नम्बर को round off में करने के लिए highest int. किया जाता है।

Syntax: math.ceil (<any number>)

Ex. print (math.ceil (3.2))

4

print (math.ceil (3))

3

print (math.ceil (-20.4))

-20

(5) Floor() → यह फंक्शन दिये गये नम्बर को lowest in round off में करता है।

Syntax. math.floor(<any number>)

Ex. print(math.floor(3.2))
3

print(math.floor(3))
3

print(math.floor(-20.4))
-21

(6) log() → यह फंक्शन दिये गये नम्बर की natural logarithm calculate करके return करता है।

Syntax math.log(<any number>)

Ex. print(math.log(2.34))
0.8501509293696l

(7) log10() → यह फंक्शन दिये गये नम्बर की 10 के लिए logarithm Calculate करके return करता है।

math.log10(<any number>)

print(math.log10(2.34))
0.3692158574101428

(8) pow() → यह फंक्शन दो argument लेता इनपुट करता है. तथा के रूप में result करता x^y है। return

Syntax. math.pow(<base>, <power>)
Ex print(math.pow(8,2))
→ 64

(9) sin() → यह फंक्शन दिये गये कोण की sin Value करता है। यह एक argument लेता है जो return में होती है। radian में बदलने के लिए "3.14/180" से Degree को radian गुणा करते हैं।

Syntax math.sin(<value in radian>)
Ex. print(math.sin(30*3.14/180))
→ .5

(10) Cos() → यह फंक्शन दिये गये कोण की cos Value करता है। यह एक argument लेता है जो return में होती है। radian में बदलने के लिए "3.14/180" से Degree को radian गुणा करते हैं।

Syntax: math.cos(<value in radian>)
Ex. print(math.cos(60*3.14/180))
→ 0.5004596890082056

(11) tan() → यह फंक्शन दिये गये कोण की tan Value करता है। यह एक argument लेता है जो return में होती है। radian में बदलने के लिए "3.14/180" से Degree को radian गुणा करते हैं।

(12) degrees( ) → यह फंक्शन को radian को degree में convert करता है।

Syntax: math.degrees(<any angle in radian>)
Example: print(math.degrees(1.0472))
→ 60.000140

(13) radians( ) → इस फंक्शन का उपयोग degree के मान को radian में बदलने के लिए किया जाता है।

(14) trunc( ) → यह फंक्शन दिए गए floating number से point के बाद की संख्या हटा देता है।

Syntax: math.trunc(<any floating point number>)
Example: print(math.trunc(6.97))
→ 6

Program: एक प्रोग्राम लिखें जिसमें x का मान इनपुट करें और निम्न समीकरण के आधार पर परिणाम की गणना करें।

```
import math
x = float(input("enter value of x"))
Cal = math.exp(x) + math.cos(math.radian(x)) +
                    math.sqrt(x)
print("Output is", Cal)
```

Program: 67
```
from math import log sqrt
x = int(input("enter value of x"))
y = int(input("enter value of y"))
Cal = log(sqrt(x/y))
print(cal)
```

Random module :—

Random number generate के लिए इसे random module के फंक्शन use किये जाते हैं।

(1) random() → 0 से 1 के बीच रेन्डम संख्या प्राप्त करने के लिए इस फंक्शन का प्रयोग होता है और यह कोई भी argument इनपुट नही करता है।

Syntax: random.random()
Example: print(random.random())
→

(2) randint() → दो इन्टीजर्स के बीच रेन्डम संख्या प्राप्त करने के लिए इसका प्रयोग होता है।

Syntax: random.randint(<start>,<end>)
Example: print(random.randint(20,25))
→ 25

String Manipulations :—

स्ट्रिंग को सिंगल, डबल या ट्रिपल quotes में लिखा जा सकता है। प्रत्येक String में character के पास index number होता है।

Forward index → 0 1 2 3 4 5
nm = "P Y T H O N"
-6 -5 -4 -3 -2 -1 ← Backward index

nm[1] → Y

Program: "PYTHON" का उपयोग एक स्ट्रिंग के रूप में करके निम्न पैटर्न उत्पन्न करने के लिए एक प्रोग्राम लिखें।

```
P     N
Y     O
T     H
H     T
O     Y
N     P
```

```
S = "PYTHON"
for a in range(0,6):
    print(S[a]," ",S[-1-a])
```

String Functions :-

(1) len → स्ट्रिंग में कैरेक्टर्स की संख्या ज्ञात करने लिए इस फंक्शन का प्रयोग होता है।

Syntax: len(<string>)
Example: S = "Python"
→ print(len(S))
→ 6

(2) ord → इस फंक्शन का उपयोग दिए गए character का ASCII कोड ज्ञात करने के लिए होता है।

Syntax: ord(<character>)
Example: print(ord('A'))
→ 65

(3) chr → इस फंक्शन का उपयोग ASCII कोड के जात करने के लिए किया जाता है। character

Syntax: chr (< any ASCII Code >)
Example: print (chr (65))
→ A

(4) str → इस फंक्शन का उपयोग फॉर्मेट को फॉर्मेट में number change करने के लिए किया जाता String है।

Syntax: str (< any no. >)
Example: print (str (123))
→ 123

String Methods :—

(1) Capitalize ( ) → इस फंक्शन का प्रयोग दी गई स्ट्रिंग के पहले या को करने के लिये होता character है। letter capital

Syntax: < string >. capitalize ( )
Example: s = "asdf"
print (s. capitalize ())
→ Asdf

(2) find ( ) → यह फंक्शन दी गई स्ट्रिंग की सब-स्ट्रिंग में (पहली उपस्थिति) का पता लगाता है। first occurrence और index position रिटर्न करता है।

Syntax: < string >. find (< sub-string >, < start >, < end >)
Example: s = "python"
print (s. find ("p"))
→ 0

(3) isalnum() → यह फंक्शन चैक करता है कि दी गई स्ट्रिंग में alpha-numeric मान है या नही। यह boolean Value रिटर्न करता है।

Syntax: <string>.isalnum()
Example: s = "@#*@"
print(s.isalnum())
→ False

(4) isalpha() → यह चैक करता है कि दी गई स्ट्रिंग में alphabet है या नही।

Syntax: <string>.isalpha()
Example: s = "Python"
print(s.isalpha())
→ True

(5) isdigit() → यह चैक करता है कि दी गई स्ट्रिंग में numbers है या नही।

Syntax: <string>.isdigit()
Example: s = "Python"
print(s.isdigit())
→ False

(6) islower() → यह चैक करता है कि दी गई स्ट्रिंग में Small letters लिखी गई है या नही।

Syntax: <string>.islower()
Example: s = "upper"
print(s.islower())
→ True

(7) isupper → यह चैक करता है कि दी गई स्ट्रिंग में लिखी गई है या नही। capital letters

Syntax: <string>.isupper()
Example: s = "UPPER"
print(s.isupper())
→ True

(8) isspace() → यह फंक्शन चैक करता है कि दी गई स्ट्रिंग में Space है या नही।

Syntax: <string>.isspace()
Example: s = " "
print(s.isspace())
→ True

(9) lower() → यह फंक्शन capital letters को small letter में बदल देता है।

Syntax: <string>.lower()
Example: s = "UPPER"
print(s.lower())
→ upper

(10) upper() → यह फंक्शन small letters को capital letters में बदल देता है।

Syntax: <string>.upper()
Example: s = "upper"
print(s.upper())
→ UPPER

(11) lstrip() → इस फंक्शन का प्रयोग दी गई स्ट्रिंग में से अग्रणी leading blank space हटाने के लिए किया जाता है।
यह फंक्शन दी गई स्ट्रिंग में से leading characters को भी हटा सकता है।

Syntax: <string>.lstrip(<character>)

Example: 1. S = " India"
print(s.lstrip())
→ India

2. p = "computer"
print(p.lstrip("com"))
→ puter

3. k = "programming"
print(k.lstrip("grop"))
→ amming

Program: शब्दों की संख्या गिनने के लिए प्रोग्राम लिखे।

```
c = 0
p = input("enter sentence")
for a in p:
    if a.isspace():
        c = c+1
print("word is", c+1)
```

(12) rstrip() → इसका प्रयोग दी गई स्ट्रिंग में से trailing blank space को हटाने के लिए किया जाता है।
इसके द्वारा स्ट्रिंग में से trailing characters भी हटाये जा सकते हैं।

Example: S = "India "
p = "Python"

```
print(s.rstrip())
print(p.rstrip('n'))
```
→ India
→ Pytho

(13) **strip()** → यह फंक्शन तथा में दिये गये कैरेक्टर को start हटा end देता है। यदि में argument character नही बताया गया तब यह space हटा देता है।

Syntax: <string>.strip()
Example: s = "☆☆☆ Python ☆☆☆"
```
print(s.strip('☆'))
```
→ Python

(14) **split()** → यह फंक्शन दी गई स्ट्रिंग को दिए गए कैरेक्टर के अनुसार अलग-अलग विभाजित कर देता है तथा यह सब-स्ट्रिंग की लिस्ट रिटर्न करेगा।

Syntax: <string>.split(<delimiter>, <num>)
<character>
Example:
```
s = 'Honesty is the best policy'
print(s.split(' '))
```
→ ['Honesty', 'is', 'the', 'best', 'policy']

```
s2 = 'Honesty is the best is policy'
print(s2.split('is', 1))
```
→ ['Honesty', 'the best is policy']

(15) **join()** → यह फंक्शन argument के रूप मे दिए गए Sequence को स्ट्रिंग के बीच मे जोड़ देता है।

Syntax: string.join(sequence)

Example: '--'.join('python')

→ 'p--y--t--h--o--n'

'..'.join(('abc', 'xyz', 'pqr'))

→ 'abc..xyz..pqr'

(16) **encode()** → यह फंक्शन दी गई स्ट्रिंग को codes में convert कर देता है।

यह फंक्शन encoding code तथा error handler argument के रूप मे इनपुट करता है।

Syntax: string.encode(<encoding code>, <error handler>)

Example: s = 'A simple string'

print(s.encode('utf_16', 'strict'))

utf_8

utf_8_sig

utf_32

b'\xff\xfe-A\x00\x00s\

(17) **decode()** → यह फंक्शन encode की गई स्ट्रिंग को decode करता है।

Syntax: string.decode(<encoding code>, <error handler>)

Example: print(s.decode('utf_16', 'strict'))

→ A Simple string

(18) **endswith()** → यह फंक्शन चैक करता है कि दी गई स्ट्रिंग दी गई सब-स्ट्रिंग ~~Suf~~ के साथ समाप्त होती है या नही और ये Boolean Value रिटर्न करता है।

Syntax: <string>.endswith(<suffix>,[<start>],[end])
Example: S = 'Rakesh kumar'
print(s.endswith('kumar'))
→ True

S = "Anant kumar singh"
print(s.endswith('kumar', 2, 11)
→ True

(19) startswith() → यह फंक्शन चैक करता है कि दी गई स्ट्रिंग दिये गये prefix के साथ स्टार्ट होती है या नही।

Syntax: <string>.startswith(<prefix>,[<start>],[end])
Example: S = 'Rakesh kumar'
print(s.startswith('R'))
→ True

S = "Anant kumar Singh"
print(s.startswith('a', 2, 11)
→ True

(20) partition() → यह फंक्शन दी गई स्ट्रिंग को दिये गये argument के अनुसार अलग-2 विभाजित कर देता है तथा एक टपल रिटर्न करता है। टपल के तीन भाग होते हैं -

Syntax: <string>.partition(<separator>)
Example: S = 'Python is a OOPs based language'
print(S.partition('OO'))
('Python is a ', 'OO', 'Ps based language')

Date & Time Function :-

Time & date को handle करने के लिए Python में दो module Time तथा calendar दिये गये हैं।

ये module समय को आठ संख्या की एक टपल मे स्टोर करते हैं।

Index	Field	Values
0	4-digit year	2008
1	month	1 to 12
2	Day	1 to 31
3	Hour	0 to 23
4	Minute	0 to 59
5	Second	0 to 60 (61 - leap Second)
6	Day of week	0 to 6 (0 - monday)
7	Day of year	1 to 366

'time' Module' :-

(1) time() → 12:00 am, January 1, 1970 के बाद वर्तमान समय तक बीते गये सैकण्ड की संख्या यह फंक्शन रिटर्न करता है।

Syntax: time.time()

Example:
```
import time
t = time.time()
print(t)
```
→ 1559938519

(2) localtime ( ) → इस फंक्शन का प्रयोग एक अन्तराल में बीते गये सेकण्ड की संख्या को time tuple में बदलने के लिए प्रयोग किया जाता है।

Syntax: time.localtime (<no. of seconds>)
Example: import time
t = time.localtime (time.time())
print ("Current time is", t)
print ("Current month is", t[1])

(3) ctime ( ) → यह फंक्शन current date & time को समझने योग्य फार्मेट में दिखाता है।

Syntax: time.ctime ( )
Example: import time
print (time.ctime ( ))
→ Fri Dec 17 10:30:00 2021

(4) mktime ( ) → यह फंक्शन Time tuple को argument के रूप में इनपुट करता है तथा एक अवधि में बीते गये सेकण्ड की संख्या को रिटर्न करता है।

Syntax: time.mktime (<time in tuple>)
Example: import time
print (time.mktime (time.local time ()))

(5) Sleep ( ) :→ यह फंक्शन दिये गये सेकण्ड तक प्रोग्राम के execution को रोक देता है।

Syntax: time.sleep (<no. of seconds>)
Example: import time
time.sleep (5)

(6) **strftime()** → यह फंक्शन फार्मेट आरग्यूमेन्ट के द्वारा time को स्ट्रिंग के रूप में बदलक argument रिटर्न करता है। यदि time tuple नही दी गई है तो समय का use करेगा। local time से वर्तमान

Syntax: time.strftime('<format>', <time tuple>)

%a	Weekday name in Short
%A	full weekday name
%b	short month name
%B	full month name
%m	(1-12) Month
%d	(1-31) Day
%D	'%m/%d/%y' formate
%H	(0-23) hour
%I	(1-12) Hour
%j	(01-366) Day of year
%n	new line character
%p	am or pm
%S	Second
%t	tab character
%T	%H:%M:%S
%u	(1-7) weekday no. (1-Monday)
~~%w~~	
%Y	Year

Example: import time
t = (2009, 2, 17, 17, 3, 38, 1, 48)
print (time.strftime('%b %d %Y %H:%M:%S', t)
→ Feb 17 2009 17:03:38

(7) strptime() → यह फंक्शन एक टाइम तथा फार्मेट के रूप में इनपुट करता है तथा इसके अनुसार argument एक time tuple रिटर्न करता है।

Syntax: time.strptime(<time>, <format>)
Example: import time
print(time.strptime('30 Aug 1969', '%d %b %Y'))

'Calendar' Module :-

(1) Calendar() → यह फंक्शन दिये गये year के लिए multi line string Calendar रिटर्न करता है।
Calendar तीन कॉलम में फार्मेट किया जाता है

Syntax: Calendar.Calendar(<year>, <w>, <l>, <c>)
w → प्रत्येक कैरेक्टर के लिए चौड़ाई है
l → प्रत्येक date सप्ताह के लिए लाइनों की संख्या है
c → कॉलम के बीच का space है

Example: import Calendar
print(Calendar.Calendar(2021, w=3, l=1, c=6)

(2) isleap() → यह फंक्शन चैक करता है कि दिया गया year leap year है या नही तथा यह boolean value रिटर्न करता है।

Syntax: Calendar.isleap(<year>)
Example: import Calendar
print(Calendar.isleap(2020))
→ True

(3) leapdays() → यह फंक्शन दिये गये years के बीच leapdays calculate करके रिटर्न करता है।

Syntax: Calendar.leapdays(<year1>, <year2>)
Example: import calendar
print(Calendar.leapdays(2000, 2020)
→ 5

(4) month() → यह फंक्शन year तथा month के रूप में दिये गये argument के अनुसार एक month Calendar रिटर्न करता है।

Syntax: Calendar.month(<year>, <month>, <w>, <l>)
Example: import calendar
print(calendar.month(2021, 12, w=2, l=1)

(5) month Calendar → यह फंक्शन दिये गये year तथा month के अनुसार month रिटर्न करता है। nested list

Syntax: calendar.month calendar(year, month)
Example: print(Calendar.month calendar(2022, 1)

(6) monthrange() → यह फंक्शन दिये गये year तथा month के अनुसार दो integer return करता है। जिसमे पहला महीने के पहले दिन के लिए weekday code होता है तथा दूसरा महीने मे दिनो की संख्या है।

Syntax: Calendar.monthrange(year, month)
Example: print(calendar.monthrange(2019,8)
→ (3,31)

(7) setfirstweekday() → इस फंक्शन के द्वारा सप्ताह की शुरुआती दिन को निर्धारित किया जा सकता है।

Syntax: calendar.setfirstweekday(<weekday code>)
Example: import calendar
calendar.setfirstweekday(5)
calendar.prmonth(2019,6)

Weekday code में 0 (Monday) तथा 6 (Sunday) है।

(8) weekday() → यह फंक्शन दी गई date में weekday code रिटर्न करता है।

Syntax: calendar.weekday(year, month, day)
Example: print(calendar.weekday(2019,4,12)