

Chapter 15
Python
Modules

Computer Science

Class XI (As per
CBSE Board)

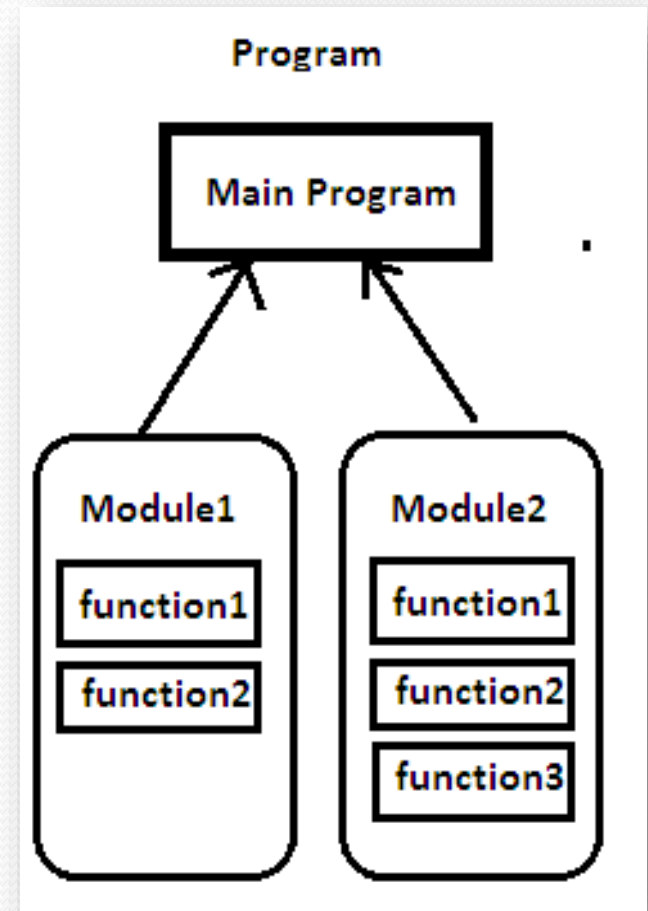
Visit : python.mykvs.in for regular updates

Python Module

A module is a logical organization of Python code. Related code are grouped into a module which makes the code easier to understand and use. Any python module is an object with different attributes which can be bind and referenced.

Simply, it is a file containing a set of functions which can be included in our application.

Python provide inbuilt standard



import - Python modules can get access to code from another module by importing the file/function using **import**.

```
import math  
print(math.pi)
```

from - pi as a whole can be imported into our initial code, rather than importing the whole module.

```
from math import pi  
print(pi)
```

In the above code module, math is not imported, rather just pi has been imported as a variable.

All the functions and constants can be imported using *****.

```
from module_name import *  
print(pi)  
print(factorial(6))
```

math module

The math module is a standard module in Python and is always available. To use mathematical functions under this module, we have to import the module using import math statement.

How to use math function

```
import math  
math.sqrt(4)
```

math.sqrt()

The **math.sqrt()** method returns the square root of a given number.

```
>>>math.sqrt(100)
```

```
10.0
```

```
>>>math.sqrt(3)
```

```
1.7320508075688772
```

The **ceil()** function approximates the given number to the smallest integer, greater than or equal to the given floating point number. The **floor()** function returns the largest integer less than or equal to the given number.

```
>>>math.ceil(4.5867)
```

```
5
```

```
>>>math.floor(4.5687)
```

```
4
```

math.pow()

The **math.pow()** method receives two float arguments, raises the first to the second and returns the result. In other words, **pow(2,3)** is equivalent to **2**3**.

```
>>>math.pow(2,4)
```

```
16.0
```

math.fabs()

Returns the absolute value of x

```
>>> import math
>>> math.fabs(-5.5)
5.5
```

The math module contains functions for calculating various trigonometric ratios for a given angle. The functions (**sin**, **cos**, **tan**, **etc.**) need the angle in radians as an argument.

```
>>> math.sin(270)
-0.1760459464712114
```

Random Module

The random module provides access to functions that support many operations. Perhaps the most important thing is that it allows us to generate random numbers.

random.randint()

Randint accepts two parameters: a lowest and a highest number.

```
import random  
print (random.randint(0, 5))
```

This will output either 1, 2, 3, 4 or 5.

random.random()

Generate random number from 0.01 to 1. If we want a larger number, we can multiply it.

```
import random  
print(random.random() * 100)
```

randrange()

generate random numbers from a specified range and also allowing rooms for steps to be included.

Syntax :

```
random.randrange(start(opt),stop,step(opt))
```

```
import random
```

```
# Using randrange() to generate numbers from 0-100
```

```
print ("Random number from 0-100 is :",end="")
```

```
print (random.randrange(100))
```

```
# Using randrange() to generate numbers from 50-100
```

```
print ("Random number from 50-100 is :",end="")
```

```
print (random.randrange(50,100))
```

```
# Using randrange() to generate numbers from 50-100
```

```
# skipping 5
```

```
print ("Random number from 50-100 skip 5 is :",end="")
```

```
print (random.randrange(50,100,5))
```

OUTPUT

Random number from 0-100 is : 27

Random number from 50-100 is : 48

Random number from 50-100 skip 5 is : 80

statistics module

This module provides functions for calculating mathematical statistics of numeric (Real-valued) data.

statistics.mean(data)

Return the sample arithmetic mean of data which can be a sequence or iterator. The arithmetic mean is the sum of the data divided by the number of data points (AVERAGE).

```
import statistics
print(statistics.mean([5,3,2]))
```

OUTPUT

3.3333333333333335

statistics.median(data)

Return the median (middle value) of numeric data, using the common “mean of middle two” method. If data is empty, `StatisticsError` is raised.

```
import statistics
print(statistics.median([5,5,4,4,3,3,2,2]))
```

OUTPUT

3.5

Visit : python.mykvs.in for regular updates

statistics.mode(data)

Return the most common data point from discrete or nominal data. The mode (when it exists) is the most typical value, and is a robust measure of central location. If data is empty, or if there is not exactly one most common value, `StatisticsError` is raised.

```
import statistics  
print(statistics.mode([1, 1, 2, 3, 3, 3, 3, 4]))
```

OUTPUT

3