CHAPTER 04

# Introduction to Python Modules

## In this Chapter...

- Importing Modules in a Python Program
- Mathematical Functions
- Random Module/Functions
- Statistics Module

A Python module can be defined as a Python program file which contains a Python code including Python functions, class or variables.

In other words, we can say that our Python code file saved with the extension (.py) is treated as the module. We have a runnable code inside the Python module.

In Python, modules provide us the flexibility to organise the code in a logical way.

## Structure of a Python Module

Structure of a Python module consists of different terms, which are as follows

- **doc string** It is useful for documentation purposes and always kept in triple quotes.
- **Variables** It is used for labels of data that are consist in program.
- **Classes** These are essentially a template to create objects.
- **Objects** These are instances of classes. Object is simply a collection of data (variables) and methods (functions).
- **Statements** These are logical instructions which can read and executed by Python interpreter.
- **Functions** It is a self-contained block of statements which are kept together to perform a specific task in related manner.

## Importing Modules in a Python Program

To make use of the function in a module, you will need to import the module with an 'import' statement.

Once we import a module, we can directly use all the functions of that module.

**Syntax** `import module_name`

This gives us access to all the functions in the module(s). To call a function of a module, the function name should be preceded with the name of the module with a dot (.) as a separator.

In Python, import statement can be used in two forms, which are as follows

### 1. To Import Entire Module

To import entire module, import statement is used. import statement is also used to import selecting modules.

**Syntax** `import module_name`

When we import a module, we are making it available to us in our current program as a separate namespace. This means that we will have to refer to the function in dot notation, as

```
module_name.function_name
```

e.g.
```
import math
print (math.sqrt (16))
```

**Output**

4.0

Here, **math** is module which contains the function definition, variables, constants etc., related to mathematical functions.

**sqrt** () is a function which will find the square root of number that given its in parentheses ().

### 2. To Import Selected Objects from a Module

When you import modules this way, you can refer to the functions by name rather than through dot notation.

**Syntax** `from module_name import object_name`

e.g.
```
from math import sqrt
print (sqrt(16))
```

**Output**

4.0

Here, **math** is module and **sqrt** () is a function. Now, we do not need to use math module with function name, as math.sqrt().

e.g.
```
from random import randint
for i in range (5):
    print (randint (1, 20))
```

**Output**

4
10
3
7
9

### To Import Multiple Objects

We can also import multiple objects in a single line. To import multiple objects, we can write the multiple objects or functions name using the comma (,) operator.

e.g. `from math import sqrt, pi`

### To Import All Objects of a Module

When you want to import all objects, you can use asterisk (*) symbol at last of keyword import.

**Syntax** `from module_name import *`

e.g. `from math import *`

## Processing of import <module> Command

When import <module> is issued, following things take place internally

- The code of module which imported is interpreted and executed.
- When module is imported, functions and variables defined in that module are now available to the program.
- To import module, same name as module a new namespace is setup.

e.g. If you imported math module in your program, all objects and attributes of that module would be referred as

`math.<object_name>`

## Processing of from <module> import <object> Command

When from <module> import <object> command is issued, following things take place internally

- The code of module which imported is interpreted and executed.
- When module is imported, only mentioned functions and variables are available to the program.
- In the current namespace, the imported definition is added because no new namespace is setup.

# Mathematical Functions

In Python, different number of mathematical operations or functions can be performed by importing the math module which contains the definition of these functions.

Some of the most popular mathematical functions are defined in math module as follows

## (i) sqrt ()

This function is used to find the square root of a specified expression or an individual number.

**Syntax**
```
math.sqrt(number)
```

*For example,*
```
>>>import math
>>>s = math.sqrt(4)
>>>print(s)
2.0
>>>s1 = math.sqrt (15)
>>>print(s1)
3.872983346207417
>>>print(math.sqrt (0))
0.0
>>>print(math.sqrt (4.5))
2.1213203435596424
```

If you give negative number as argument it will give an error.
```
>>>s2 = –4
>>>print(math.sqrt(s2))
Trackback (most recent call last):
File "<pyshell#14>", line1,in<module>
   print(math.sqrt (s2))
ValueError math domain error
```

## (ii) ceil()

This method returns ceiling value of x i.e. the smallest integer not less than x.

**Syntax**
```
math.ceil(x)
```

*For example,*
```
>>>import math
>>>c = math.ceil(45.23)
>>>print(c)
46
>>>c1 = math.ceil(–76.89)
>>>print(c1)
-76
>>>a = 456.14
>>>print(math.ceil(a))
457
>>>x = math.pi
>>>print(math.ceil(x))
4
```

## (iii) floor()

This method is used to return a value which is less than or equal to a specific expression or value.

**Syntax**

```
math.floor(x)
```

*For example,*

```
>>>import math
>>>f = math.floor (145.35)
>>>print(f)
145
>>>a = 102.78
>>>print(math.floor(a))
102
>>>b = –75.50
>>>print(math.floor(b))
–75
>>>x = math.pi
>>>print(math.floor(x))
3
```

## (iv) pow()

This method offers to compute the power of a number and hence can make task of calculating power of a number easier. In this, two types to calculate power.

- **pow** (x, y) converts its arguments into float and then computes the power.

**Syntax**

```
pow(x,y)
```

*For example,*

```
>>>import math
>>>x = 3
>>>y = 4
>>>print(pow(y, x))
64
>>>print(pow(7, 2))
49
>>>print(pow(-3, 2))
9
>>>print(pow(-4, 3))
-64
>>>print(pow(5, -2))
0.04
```

- **pow (x, y, mod)** converts its arguments into float and then computes the power.

**Syntax**

```
pow(x, y, mod)
```

*For example,*

```
>>>x = 4
>>>y = 3
>>>z = 10
>>>pow(x, y, z)
4
>>>pow(5, 3, 7)
6
>>>print(pow(6, 0, 2))
1
>>>pow(0, 4, 2)
0
>>>pow (8, 3, 0)
Trackback (most recent call last):
File "<pyshell#7>", line 1, in <module>
   pow(8, 3, 0)
ValueError : pow() 3rd argument cannot be 0.
```

## (v) fabs()

This method returns the absolute value (positive value) of x.

**Syntax**

```
math.fabs(x)
```

*For example,*

```
>>>import math
>>>x = -25
>>>math.fabs (x)
25.0
>>>print(math.fabs (65))
65.0
>>>print(math.fabs(-4.3))
4.3
```

## (vi) sin()

This method returns the sine of value passed as argument. The value passed in this function should be in radians.

**Syntax**

```
math.sin(x)
```

*For example,*

```
>>>import math
>>>x = 65
>>>math.sin(x)
0.8268286794901034
>>>print(math.sin(5.3245521))
-0.8184069203707078
>>>a = math.pi
>>>x = a/4
>>>math.sin(x)
0.7071067811865475
```

## (vii) cos()

This method returns the cosine of value passed as argument. The value passed in this function should be in radians.

**Syntax**

```
math.cos(x)
```

*For example,*

```
>>>import math
>>>x=9
>>>math.cos(x)
-0.9111302618846769
```

```
>>>math.cos(0)
1.0
>>>print(math.cos(30))
0.15425144988758405
>>>math.cos(-4)
-0.6536436208636119
>>>a = math.pi
>>>math.cos(a)/2
-0.5
```

### (viii) tan()

This method returns the tangent of value passed as argument. The value passed in this function should be in radians.

**Syntax**

```
math.tan(x)
```

*For example,*

```
>>>import math
>>>x=30
>>>math.tan(x)
-6.405331196646276
>>>math.tan(0)
0.0
>>>math.tan(90)
-1.995200412208242
>>>print(math.tan(-5))
3.380515006246586
```

### (ix) pi

It is a mathematical constant, the ratio of circumference of a circle to its diameter.

**Syntax**

```
math.pi
```

*For example,*

```
>>>import math
>>>math.pi
3.1415926.....
```

### (x) e

It is a mathematical constant.

**Syntax**

```
math.e
```

*For example,*

```
>>>import math
>>>math.e
2.71828182846
```

## Random Module/Functions

Python offers random module that can generate random numbers. These random modules depend on a pseudo random number that generate function random() and this number generates a random float number between 0.0 and 1.0.

Some random functions are as follows

### (i) random()

This method is used to generate a float random number less than 1 and greater than or equal to 0. This function does not require any arguments.

**Syntax**

```
random.random()
```

*For example,*

```
>>>import random
>>>random.random()
0.7358047613841759
```

### (ii) randint ()

This method is one of methods that handle random numbers. It has two parameters start and end and generate an integer between start and end (including both).

**Syntax**

```
random.randint(start, end)
```

*For example,*

```
>>>import random
>>>random.randint(10, 50)
34
>>>print(random.randint(5, 70))
19
>>>random.randint(-80, -20)
-20
>>>random.randint(-12, 60)
35
```

### (iii) randrange()

This method returns a random selected element from the range created by the start, stop and step arguments. The value of start is 0 by default. Similarly, the value of step is 1 by default.

**Syntax**

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

*For example,*

```
>>>import random
>>>random.randrange (10, 100, 5)
70
>>>random.randrange(20, 30, 10)
20
>>>random.randrange(-50, -20, 10)
-30
>>>random.randrange(-10, 0, 4)
-6
```

### (iv) choice()

This method is used to generate 1 random number from a container. An empty sequence as argument raises an IndexError.

**Syntax**

```
random.choice (seq/list/tuple/dictionary)
```

*For example,*

```
>>>import random
>>>random.choice('Programming')
'm'
>>>random.choice([12, 74, 32, 65, 0, 23])
0
>>>random.choice ([12, 74, 32, 65, 0, 23])
65
>>>random.choice ([78, 90, 43, 32, 67])
32
>>>random.choice ({1 : 'One', 2 : 'Two', 3 : 'Three', 4
: 'Four'})
'One'
```

### (v) shuffle()

This method randomly reorder the elements in a list. It can shuffle only list elements.

**Syntax**

```
random.shuffle(list)
```

*For example,*

```
>>>import random
>>>num = [78, 54, 89, 55, 65, 12]
>>>random.shuffle(num)
>>>num
[78, 89, 12, 54, 65, 55]
>>>random.shuffle(num)
>>>num
[55, 89, 54, 65, 12, 78]
```

## Statistics Module

Python is a very popular language when it comes to data analysis and statistics. Python has ability to solve the mathematical expression, statistical data by importing statistics keyword. Statistics module was added in Python 3.4 version. Earlier version of Python cannot access this module. To access Python's statistics functions, we need to import the functions from the statistics module.

Some statistics functions are as follows

### (i) mean()

It returns the simple arithmetic mean of data which can be a sequence or iterator. Arithmetic mean is the sum of data divided by the number of data set.

**Syntax**

```
statistics.mean(data_set)
```

*For example,*

```
>>>import statistics
>>>list=[45,78,21,32,45,56]
>>>statistics.mean(list)
46.166666666666664
>>>list=[-14,25,-32,-12,0,65]
>>>statistics.mean(list1)
5.333333333333333
>>>t = (14,14,12,32,78)
>>>statistics.mean(t)
30.0
>>>t1=(-12,-23,45,-2,0,16)
>>>statistics.mean(t1)
4.0
>>>l1=[1.2,2.3,4.5,6,-8]
>>>statistics.mean(l1)
1.2
>>>statistics.mean()
Traceback (most recent call last):
File ''<pyshe||#11>'', line 1, in <module>
   statistics.mean()
TypeError:mean()missing 1 required positional
argument:'data'
```

### (ii) median()

This function calculates middle value of the arithmetic data in iterative order. If there are an odd number of values, median() returns the middle value. If there are an even number of values it returns an average of two middle values.

**Syntax**

```
statistics.median(data_set)
```

*For example,*

```
>>>import statistics
>>>list=[12,54,89,65,78]
>>>statistics.median(list)
65
>>>list1=[45,-12,-65,78,0]
>>>statistics.median(list1)
0
>>>list2=[-12,4.65,78,-98,45,6.5]
>>>statistics.median(list2)
5.575
>>>t=(78,98,23,-32,8.6,-9)
>>>statistics.median(t)
15.8
```

### (iii) mode()

This function returns the number with maximum number of occurrences.

**Syntax** `statistics.mode(dataset)`

*For example,*

```
>>>import statistics
>>>list=[45,89,45,78,65,32,45,66]
>>>statistics.mode(list)
45
>>>list1=[4.5,6.5,7.0,6.5,98,4.5,6.5]
>>>statistics.mode(list1)
6.5
>>>t=(-45,-89,0,-89,36,-86,-36)
>>>statistics.mode(t)
-89
```

When two numbers have same occurrence of number then it will give first number of maximum occurrence.

```
>>>list2=[-45,-89,0,-89,36,-36,-36]
>>>statistics.mode(list2)
– 89
```

# Chapter Practice

PART 1

## Objective Questions

### • Multiple Choice Questions

**1.** Which module is used for sqrt () to find the square root of a number?

(a) random (b) square root
(c) math (d) statistics

***Ans.*** (*c*) sqrt () is used to find the square root of a specified expression or an individual number. It is performed by importing math module.

**Syntax** `math.sqrt (number)`

**2.** Identify the correct output.

```
print (pow(–3, 2))
```

(a) 9 (b) **–9**
(c) 8 (d) **– 8**

***Ans.*** (*a*) pow( ) converts its argument into float and then computes the power. First argument is the number whose power to be find. Here power is 2 which is even number which gives positive result either base number is odd or even.

**3.** Identify the correct output of

```
>>>import math
>>>f = math.floor (145.35)
>>> print (f)
```

(a) 146 (b) 145
(c) 145.3 (d) 145.4

***Ans.*** (*b*) floor () is used to return a value which is less than or equal to a specific expression or value.

**4.** What is returned by math.ceil(7.9)?

(a) 7 (b) 8
(c) 7.0 (d) 9.0

***Ans.*** (*b*) The ceil function returns the smallest integer that is bigger than or equal to the number itself.

**5.** To include the use of functions which are present in the random library, we must use the option

(a) import random (b) random.h
(c) import.random (d) random.random

***Ans.*** (*a*) The command import random is used to import the random module, which enables us to use the functions which are present in the random library.

**6.** What will be the output of the following Python function if the random module has already been imported?

```
random.randint(3.5,7)
```

(a) Error
(b) Any integer between 3.5 and 7, including 7
(c) Any integer between 3.5 and 7, excluding 7
(d) The integer closest to the mean of 3.5 and 7

***Ans.*** (*a*) The function random.randint() does not accept a decimal value as a parameter. Hence, the function shown above will throw an error.

**7.** What will be the output of the following Python code?

```
random.randrange(0,91,5)
```

(a) 10 (b) 18
(c) 79 (d) 95

***Ans.*** (*a*) The function shown above will generate an output which is a multiple of 5 and is between 0 and 91. The only option which satisfies these criteria is 10. Hence, the only possible output of this function is 10.

**8.** What will be the output of the following Python code?

```
random.randrange(1,100,10)
```

(a) 32 (b) 67
(c) 91 (d) 80

***Ans.*** (*c*) The output of this function can be any value which is a multiple of 10, plus 1. Hence a value like 11, 21, 31, 41…91 can be the output. Also, the value should necessarily be between 1 and 100. The only option which satisfies this criterion is 91.

**9.** Which extension is used to save the Python code file?

(a) .Python (b) .py (c) .p (d) .pyth

***Ans.*** (*b*) Python code file saved with the extension .py is treated as the module. Python module can be defined as Python program file which contains a Python code including Python functions, class or variables.

**10.** Which of the following symbol is used to import all objects of a module?

(a) * (b) # (c) @ (d) $

***Ans.*** (*a*) When you want to import all objects, you can use asterisk (*) symbol at last of keyword import.

**Syntax**

```
from module_name import*
```

## • Case Based MCQs

**11.** A Python module is a file containing Python definitions and statements. A module can define functions, classes and variables. A module can also include runnable code. Grouping related code into a module makes the code easier to understand and use. It also makes the code logically organised. When you import a module, the Python interpreter searches for the module in the following sequences.

- The current directory.
- If the module isn't found, Python then searches each directory in the shell variable PYTHONPATH.
- If all else fails Python checks the default path. On UNIX, this default path is normally/usr/local/lib/python/.

(i) To import multiple objects, which symbol is used to write multiple objects?

(a) # (b) :
(c) ; (d) ,

(ii) Which of the following module is used in Python?

(a) math (b) pie
(c) statistics (d) square root

(iii) Which keyword is used to import the module?

(a) import
(b) import_module
(c) module
(d) Any of the above

(iv) Which of the following module is used for mean(), mode () and median()?

(a) math (b) arithmetic
(c) statistics (d) random

(v) Choose the correct syntax to import all objects of a module.

(a) `from module_name import *`
(b) `from module_name import all`
(c) `module_name * from import`
(d) `all module from import`

***Ans.*** (i) (*d*) We can also import multiple objects in a single line. To import multiple objects, we can write the multiple objects or functions name using the comma (,) operator.

(ii) (*a*) math module is used in Python which contains the function definition, variables, constants etc., related to mathematical functions.

(iii) (*a*) To make use of the function in a module, you will need to import the module with an 'import' statement.

**Syntax** `import module_name`

(iv) (*c*) statistics module is used for mean (), mode () and median (). statistics module was added in Python 3.4 version. Earlier version of Python cannot access this module. To access Python's statistics functions, we need to import the functions from the statistics module.

(v) (*a*) When you want to import all objects, you can use asterisk (*) symbol as last of keyword import.

**Syntax** `from module_name import*`

## • Short Answer Type Questions

**1.** What will be the output of the given code?

```
import random
random.randrange (1, 50, 10)
```

***Ans.*** The output of this code can be any value which is a multiple of 10, plus 1. Hence, a variable like 11, 21, 31, 41 can be output. Also, the value should necessarily be between 1 and 50.

**2.** What is return by following code?

(i) `math.ceil(8.7)`
(ii) `math.floor(9.5)`

***Ans.*** (i) 9 (ii) 9

**3.** What is the output of the given code?

```
from random import shuffle
x = ['One', 'Two', 'Three', 'Four', 'Five',
                                    'Six']
shuffle(x)
print(x)
```

***Ans.*** **Output** ['Four', 'Five', 'Two', 'Six', 'One', 'Three']

**4.** Write the short note on

(i) ceil() (ii) floor()

***Ans.*** (i) **ceil()** method returns ceiling value of x i.e, the smallest integer not less than x.

**Syntax** `math.ceil(x)`

(ii) **floor()** method is used to return a value which is less than or equal to a specific expression or value.

**Syntax** `math.floor(x)`

**5.** What is the output of the given code?

(i) `int (math.pow (5, 3))`
(ii) `math.pow(5, 3)`
(iii) `int (pow(5, 3, 4))`

***Ans.*** (i) 125 (ii) 125.0 (iii) 1

**6.** How can you generate random numbers in Python?

***Ans.*** random module is the standard module that is used to generate a random number. The method is defined as

```
import random
random.random()
```

The statement random.random() returns the floating point number that is in the range of (0, 1). The function generates random float numbers. The methods that are used with the random class are the bound methods of the hidden instances.

**7.** Define the sqrt() method with an example.

***Ans.*** sqrt() method is used to find the square root of a specified expression or an individual number, math module is used to import this function.

**Syntax**

```
import math
math.sqrt(number)
```

e.g.

```
>>>import math
>>> math.sqrt(25)
5.0
>>> math.sqrt(16.9)
4.110960958218893
```

**8.** How to import entire module in Python?

***Ans.*** To import entire module, import statement is used. import statement is also used to import selecting modules.

**Syntax** `import module_name`

When we import a module, we are making it available to us in our current program as a separate namespace.

**9.** What things take place internally, when from <module> import <object> command is used?

***Ans.***
- The code of module which imported is interpreted and executed.
- When module is imported, only mentioned functions and variables are available to the program.
- In the current namespace, the imported definition is added because no new namespace is setup.

**10.** Distinguish between floor() and ceil().

***Ans.*** Differences between floor() and ceil() are as follows

| **floor**() | **ceil**() |
|---|---|
| It accepts a number with decimal as parameter and returns the integer which is smaller than the number itself. | It accepts a number with decimal as parameter and returns the integer which is greater than the number itself. |
| **Syntax** | **Syntax** |
| `math.floor()` | `math.ceil ()` |

## • Long Answer Type Questions

**11.** Predict the output.

```
import math
print ("cos:", math.cos(1.047197551))
print ("sin:", math.sin(0.523598775))
print ("tan:", math.tan(0.463647609))
print ("degree:", math.degrees(3.1415926))
print ("radian:", math.radians(180))
```

***Ans.*** **Output**

cos : 0.5000000001702586
sin : 0.4999999994818579
tan : 0.49999999999899236
degree : 179.99999692953102
radian : 3.1415926535489793

**12.** What is the output of the given code?

```
import statistics
from fractions import fraction as F
from decimal import decimal as D
a = statistics.mean ([11, 2, 13, 14, 44])
b = statistics.mean ([F(8, 10), F(11, 20), F(2,
                                          5), F(28, 5)])
c = statistics.mean ([D("1.5"), D("5.75"),
                                  D("10.625"), D("2.375")])
print ("Simple mean:", a)
print ("Fraction mean:", b)
print ("Decimal mean:", c)
```

***Ans.*** **Output**

Simple mean : 16.8
Fraction mean: 147/80
Decimal mean: 5.0625

**13.** Predict the output.

```
import statistics
list = [5, 2, 5, 6, 1, 2, 6, 7, 2, 6, 3, 5, 5]
x = statistics.mean (list)
print (x)
y = statistics.median (list)
print (y)
z = statistics.mode (list)
print (z)
```

***Ans.*** **Output**

4.230769230769231
5
5

**14.** Find the output of the following code.

```
import math
print ("ceil:", math.ceil (5.24))
print ("fabs:", math.fabs (5.24))
print ("fabs:", math.fabs (-5.24))
print ("floor:", math.floor (-5.24))
print ("pow:", math.pow (3, 5))
print ("round:", round (3.14159))
print ("round:", round (3.14159,3))
print ("sqrt:", math.sqrt (64))
```

***Ans.*** **Output**

ceil: 6
fabs: 5.24
fabs: 5.24
floor: – 6
pow: 243.0
round: 3
round: 3.142
sqrt: 8.0

**15.** Define the random module in Python.

***Ans.*** Python offers random module that can generate random numbers. There are various types of random functions which can import by random keyword.
Some of them are describe below

(i) **random**() This method is used to generate a float random number less than 1 and greater than or equal to 0. It does not require any parameters.

**Syntax** `random.random()`

(ii) **randint()** This method is one of methods that handles random numbers. It has two parameters start and end generate an integer between start and end (including both).

**Syntax** `random.randint(start, end)`

(iii) **randrange()** This method returns a random selected element from the range created by the start, stop and step arguments. By default, the value of start is 0 and the value of step is 1.

**Syntax** `random.randrange(start, stop, step)`

(iv) **choice()** This method is used to generate 1 random number from a container.

**Syntax** `random.choice(sequence)`

(v) **shuffle()** This method randomly reorder the elements in a list. It can shuffle only list elements.

**Syntax** `random.shuffle(list)`

**16.** Modules refer to a file containing Python statements and definitions. A file containing Python code, for example : example.py, is called a module, and its module name would be example. We use modules to break down large programs into small manageable and organised files. Furthermore, modules provide reusability of code. Module focuses on small proportion of the problem, rather than focusing on the entire problem.

(i) The output of the following code is either 1 or 2. State whether this statement is true or false.

```
import random
random.randint (1, 2)
```

(ii) What is module in Python?

(iii) Which function is equivalent to random. randint (4, 7)?

(iv) What is return by math.floor (–20.0)?

(v) How to import modules in Python?

***Ans.*** (i) True

(ii) Modules can define functions that you can reference in other Python files.

(iii) 4 + random. randrange (4) on return any one of 4, 5, 6 and 7.

(iv) 20

(v) Modules can be imported using the import keyword.

**17.** Write a program to input any two matrices and print sum of matrices. **[NCERT]**

***Ans.***

```
import random
m1 = int(input ("Enter total number of rows in the first matrix"))
n1 = int(input ("Enter total number of columns in the first matrix"))
a = [[random.random() for row in range(m1)] for col in range (n1)]
for i in range (m1):
   for j in range (n1):
      a[i][j] = int(input())
m2 = int(input ("Enter total number of rows in the second matrix"))
n2 = int(input ("Enter total number of columns in the second matrix"))
b = [[random.random () for row in range (m2)] for col in range (n2)]
for i in range (m2):
   for j in range (n2):
      b [i][j] = int(input ())
c = [[random.random () for row in range (m1)] for col in range (n1)]
if ((m1 == m2) and (n1 == n2)):
   print("Output is")
   for i in range (m1):
      for j in range (n1):
         c[i][j] = a[i][j] + b[i][j]
   for s in c:
      print(s)
else:
   print("Matrix addition is not possible")
```

**18.** Write a program to input any two matrices and print product of matrices. **[NCERT]**

***Ans.***

```
import random
m1 = int(input ("Enter number of rows in first matrix"))
n1 = int(input ("Enter number of columns in first matrix"))
a = [[random.random () for row in range (m1)] for col in range (n1)]
for i in range (m1):
   for j in range (n1):
      a[i][j] = int(input ())
m2 = int(input ("Enter the number of rows in the second matrix"))
n2 = int(input ("Enter the number of columns in the second matrix"))
b = [[random.random () for row in range (m2)] for col in range (n2)]
for i in range (m2):
   for j in range (n2):
      b[i][j] = int(input ())
c = [[random.random () for row in range (m1)] for col in range (n2)]
if (n1 == m2):
   for i in range (m1):
      for j in range (n2):
         c[i][j] = 0
         for k in range (n1):
            c[i][j] += a[i][j]*b[i][j]
   for s in c:
      print(s)
else:
   print("Multiplication is not possible")
```

# Chapter Test

## Multiple Choice Questions

**1.** Which module is used for pow() to find the power of a number?
(a) random (b) math
(c) statistics (d) power

**2.** Identify the correct output of following code.
```
import math
print (math.floor (153.42))
```
(a) 153 (b) 154
(c) 154.0 (d) 153.4

**3.** To include the use of functions which are present in the statistics library, we must use the option
(a) statistics.h (b) import statistics
(c) import.statistics (d) statistics.statistics

**4.** The value passed in sin() should be in
(a) degree (b) meter
(c) inch (d) radian

**5.** Which of the following function always gives output in integer form?
(a) random () (b) choice()
(c) mean() (d) randint()

## Short Answer Type Questions

**6.** What is the output of following code?
```
import math
print(int (math.pow (4, 2)))
print (math.pow (4, 2))
print (math.ceil (4.23))
```

**7.** What is the output of following code?
```
math.ceil (9.6)
math.floor (9.4)
math.floor (-9.4)
```

**8.** Identify the output of following code.
```
import math
print ('cos:', math.cos (42.3651))
print ('sin:', math.sin (1))
print ('tan:', math.tan (0))
```

**9.** Write a short note on
(i) random()
(ii) randint()

**10.** Distinguish between mean() and mode().

## Long Answer Type Questions

**11.** What will be the output of following code?
```
import math
print ('ceil:', math.ceil (8.65))
print ('fabs:', math.fabs (8.65))
print ('fabs:', math.fabs (-8.65))
print ('floor:', math.floor (-8.65))
print ('pow:', math.pow (5, 4))
print ('round:', round (7.654265))
print ('round:', round (7.654265, 2))
print ('sqrt:', math.sqrt (289))
```

**12.** What is the output of following code?
```
import statistics
list1 = [23, 45, 3, 5, 6, 7, 12, 32, 11, 22,
                                                8, 45]
a = statistics. mean (list1)
print ("Mean is:", a)
b = statistics.median (list1)
print ("Median is:",b)
c = statistics.mode (list1)
print ("Mode is:",c)
```

**13.** Identify the output of following code.
```
import statistics
from fractions import Fraction as F
from decimal import Decimal as D
a = statistics.mean ([45,65,22,78,65,23,99])
b = statistics.mean ([F(8, 10), F(11, 20), F (2, 5), F
(28, 5)])
c = statistics.mean ([D ('1.5'), D ('5.75'), D
('10.625'), D('2.375')])
print ('Simple mean:', a)
print ('Fraction mean:', b)
print ('Decimal mean:', c)
```

# Answers

**Multiple Choice Questions**

***1. (b) 2. (a) 3. (b) 4. (d) 5. (d)***