

## CHAPTER 1: Basics of Python Programming

**Q1.** for i in range  
(100,50, -10): print(i)

```
100
90
80
70
60
>>>
```

**Q2.** x = 10; y = 5;  
print(x%y);

```
python/
>>>
0
>>>
```

**Q3.** a = 10; b = 0;  
while(a>b):  
print (a, b)  
a=a-1;  
b=b+1;

```
==== KES
.py ====
10 0
9 1
8 2
7 3
6 4
>>>
```



**Q4.** for i in range (1,20):  
if i%5==0:  
break  
else:  
print(i)  
for i in range (1,20):  
if i%5==0:  
continue  
else:  
print(i)

```

while (4>5):
    print('DONE')
else:
    print ('NOT DONE')

==== RESTART: C:/P
1
2
3
4
1
2
3
4
6
7
8
9
11
12
13
14
16
17
18
19
NOT DONE
>>>

```

**Q5.** while (5\*2 > 10):  
 print ('JUST DO IT')  
 else:  
 print('DONE')

```

==== RESTART: C:/Users/aksna/AppData/L
DONE
>>>

```

**Q6.** for i in range (5):  
 print ('hello!', end= ' ')

```

==== RESTART: C:/Users/aksna/AppData/L
hello! hello! hello! hello! hello!
>>>

```

**Q7.** for i in range (10):  
 if i%2 !=0:  
 print(i+1)

```

2
4
6
8
10
>>>

```

**Q8.** for i in range (10,0, -1):  
print (i, end="")

```
==== RESTART: C:\
10987654321
>>>
```

**Q9.** for i in range (5,25,3):  
print (i, end="")

```
==== RESTART: C:\
581114172023
>>>
```

**Q10.** for i in 'PYTHON':  
print (i, '\_', 'end="")

```
==== RESTART: C:/User
P _ Y _ T _ H _ O _ N _
>>>
```

**Q11.** for i in range (10):  
pass  
print(i)

```
==== RE
9
>>>
```

**Q12.** i=1  
while i<=6:  
print(i),  
i=i+1  
print('Done')

```
1
Done
2
Done
3
Done
4
Done
5
Done
6
Done
>>>
```

**Q13.** for letter in 'HELLO':

```
Pass
///
==== RESTART: C:/Users/a
>>>
```

**Q14.** weather = 'foggy'

```

if weather=='sunny':
    print ('Take your shades')
elif weather=='raining':
    print ('Take your umbrella')
else:
    print ('Stay at home')

=====
Stay at home

```

**Q15.** num = 10

```

if num > 30:
if num > 40:
    print('Python')
else:
    print('Programming')
elif num < 20:
if (num!=0):
    print ('is fun..')
    print ('!snt it !!!')

=====
is fun..
!snt it !!!

```

**Q16.** a = 5

```

b = 70
if not 10 + 10 == b or a == 40 and 70 == 80:
    print('Yeah')
elif a! = b:
    print('Nope')

=====
Yeah

```

**Q17.** if (10 << ((41 - 40 // 2 + 1) %2)) == 60:

```

print('WoW')

===== RESTART: C:/use
>>>

```

**Q18.** if not True:

```

print('WRONG')
elif not ((10+10 - 20 * 3) == 30):
print ('Really!')
else:
print ('May Be')

===== RESTART:
Really !

```

**Q19.** if (10 == 100) and (100 + 200 > 300):

```
print ('Win Win...')
else:
print ('Oh No !!!')
```

```
==== RESTART: C:\Python36\Python36\
Oh No !!!
>>>
```

**Q20.** num = 3

```
if num == 2:
print('Yes')
elif num == 4:
print('Yes')
elif num == 8:
print('Yes')
else:
print ('Number is not an even number')
```

```
Number is not an even number
>>>
```

**Q21.** num = 3

```
if (num ** 3) > (5%2 - 3 * 4 /2):
if (num // 4) >= 2:
print ('You win')
else:
print ('Try Again')
```

```
==== RESTART: C:\Python36\Python36\
Try Again
>>>
```

**Q22.** num = int ((10\*6 + 10 - 20) > 0)

```
if num == 50:
print ('You win...')
else:
print ('Try Again')
```

```
==== RESTART:
Try Again
>>>
```

**Q23.** years = 99

```
if years > 30:
print ('30')
if years < 50:
```

```
print ('50')
if years == 7:
    print ('70')
```

```
==== RESTART: C:/Users/
30
>>>
```

**Q24.** years = 99

```
if (years == 100):
    print('Century')
elif (years == 75):
    print ('Platinum Jubilee')
elif (years == 50):
    print ('Half Century')
elif (years == 25):
    print ('Silver Jubilee')
elif (years == 10):
    print('Decade')
else:
    print ('Long way to go....')
```

```
==== RESTART: C:/Users/
Long way to go....
>>>
```

**Q25.** if (5.0 < 9.0):

```
    print('DONE')
```

```
==== RESTART: C:/Users/
DONE
>>>
```

**Q26.** a, b, c = 12,13,16

```
d = a + b *c/b
print(d)
```

```
==== RESTART: C:/Users/
28.0
>>>
```

**Q29.** print (bool (int (0)))

```
print (bool (str (0)))
```

```
print (bool (float (0.0)))
```

```
print (bool (str (0.0)))
```

```
==== RESTART: C:/Users/
False
True
False
True
>>>
```

**Q30.** a = True;

b = 0<5

a==b;

a is b

```
==== RESTART: C:/Users/aksna/AppData/L
>>>
```

**Q31.** print('Hello')

print ('My Dear Students', end='')

print ('Let us learn Python')

print('Programming')

```
==== RESTART: C:/Users/aksna/AppData/L
Hello
My Dear StudentsLet us learn Python
Programming
```

**Q32.** print ('Python Programming \n is FUN !!!!')

```
==== RESTART: C:/User
Python Programming
is FUN !!!!
```

**Q33.** print ('R It is Teacher's Day today')

```
R It is Teacher's Day today
```

Q34. str = 'I Love Programming in Python'

print (str [0])

print (str [7])

print (str [-5])

print (str [7:10])

print (str [:2])

print (str [3:11:3])

print (str [-7])

```
I
P
Y
Pro
ILv rgamn nPto
o o
I Love Programming in
```

**Q35.** x, y, z = (10,20,30)

y, z, x = x+5, y+7, z-3

print ('x = ', x,'y= ', y,'z= ', z)

```
x = 27 y= 15 z= 27
```

**Q36.** `x, y = 10,20`

`y, z = x+5, y-20`

`print ('x= ', x,'y= ', y,'z= ', z)`

```

'''
===== RESTART: C:/Use:
x= 10 y= 15 z= 0
'''

```

**Q37.** `days='Mon Tue Wed Thu Fri Sat Sun'`

`months='Jan\nFeb\nMar\nApr\nMay\nJun\nJul\nAu'`

`print ('Days are:'+ days)`

`print ('Months are:'+ months)`

`print ('There's a new dream today. I'll tell you some other day. Come on, let's enjoy.')`

```

===== RESTART: C:/Users/akshar/AppData/Local/Programs/Python/Python39/x.py =====
Days are:Mon Tue Wed Thu Fri Sat Sun
Months are:Jan
Feb
Mar
Apr
May
Jun
Jul
Au
There's a new dream today.I'll tell you some other day.Come on, let's enjoy.

```

**Q38.** `name=input ('Enter a word:')`

`print ('HELLO'+ name)`

```

'''
Enter a word :AKSHARA
HELLOAKSHARA
'''

```

**Q39.** `n1=90`

`n2=n1+50`

`n2 =int(str(n2) +'40')`

`print(n2)`

```

===== REST
14040
'''

```

**Q40.** `a, b=10,20`

`b, a=a*5, b/2`

`print ('a= ', a,'and b= ', b)`

```

'''
===== RESTART: C:/Use:
a= 10.0 and b= 50
'''

```



Q41. a, b=10,20

```
a, b, a=a*10, b*5, a*20
print ('a= ', a and 'b= ', b)

a= b= 100
~~~
```

Q42. x=10

```
print(id(x))
x=x+10
print(id(x))
x=x-10
print(id(x))

1683582256
1683582576
1683582256
```

Q43.

```
>>> N = 1 + 1 + 1 == 0.3; print(N)
False
```

Q44.

```
>>> 30 < 70 and 40 > 50
False
```

Q45.

```
>>> 2*(5*(len('007'))
30
```

Q46.

```
>>> len('abcd')==30/6 or 30/10
3.0
```

Q47.

```
>>> 70/(7-(2+4)) or 4<5
70.0
```

Q48.

```
>>> 5 < 9 or 60/(10-(6+4))
True
```

Q49.

```
>>> 'R'=='R'
True
```

Q50.

```
>>> str(10)==str('10.0')
False
```

Q51.

```
>>> 10 == int(10)
True
```

Q52.

```
>>> 20 or len(20)
20
```

Q53.

```
>>> 10 == 10.0
True
```

Q54.

```
True
>>> 23%5 is 23%5
True
```

Q55.

```
>>> bool('0') and (10 < 20)
True
```

Q56.

```
>>> len(str(7 -5 -12 > -5 *3 +8))==len('true')
False
```

Q57.

```
>>> 7 -5 -12 > -5 *3 +8
False
```

Q58.

```
>>> 10 ** 3 ** 2
1000000000
```

Q59.

```
>>> 'k' or 'r'
'k'
```

Q60.

```
>>> '' or ''
''
```

Q61.

```
>>> 0 or 0
0
```

Q62.

```
>>> "abc" or ""
'abc'
```

Q63.

```
>>> 10 or 0
10
```

Q64.

```
SyntaxError: in
>>> 0 or 10
10
```

Q65.

```
>>> x = 100;x*=3;print(x)
300
```

Q66.

```
>>> 'PYTHON' "PROGRAMMING"
'PYTHONPROGRAMMING'
```

Q67.

```
>>> 'I ENJOY PROGRAMMING IN PYTHON'
'I ENJOY PROGRAMMING IN PYTHON'
```

Q68.

```
>>> 'PYTHON'
'PYTHON'
```

Q69.

```
>>> 'PROGRAMMING IN PYTHON'
'PROGRAMMING IN PYTHON'
```

Q70.

```
>>> 'I'+'LOVE'+'PROGRAMMING'
'ILOVEPROGRAMMING'
```

Q71.

```
>>> format(10**25, '.2e')
'1.00e+25'
```

Q72.

```
>>> format(987654321.315978, ',.3f')
'987,654,321.316'
```

Q73.

```
>>> print(type(int('10')))
<class 'int'>
```

Q74.

```
>>> str(print())+'abc'
'Noneabc'
```

Q75.

```
>>> print(print('abc'))
abc
None
```

Q76.

```
>>> str(print('abc'))+'xyz'
abc
'Nonexvz'
```

Q77.

```
>>> print(print('abc',end=' '))
abcNone
```

Q78.

```
>>> 350+230-170
410
```

Q79.

```
>>> (10<20)and(20 < 10)or(5 < 30)and not(15 < 40)
False
```

Q80.

```
>>> (51+9.7-4)*10
567.0
```

Q81.

```
>>> 110%(95//3)
17
```

## CHAPTER 2: Revisiting Data Structures in Python

**Q1.** `s = 'Welcome'`

`print (s [1:3])`

```
'''  
== RESTART:  
el  
>>>
```

**Q2.** `s = 'Welcome'`

`print (s [: 6])`

```
== RESTART:  
Welcom  
>>>
```

**Q3.** `s = 'Welcome'`

`print (s [4:])`

```
== RESTART:  
ome  
>>>
```

**Q4.** `s = 'Welcome'`

`print (s [1: -1])`

```
== RESTART:  
elcom  
>>>
```

**Q5.** `str = 'Welcome'`

`print ('come' in str)`

```
== RESTART:  
True  
>>>
```

**Q6.** `str = 'Welcome'`

`print ('come' not in str)`

```
== RESTART:  
False  
>>>
```



Q7.

```
>>> 'free' == 'freedom'
False
```

Q8.

```
>>> print('12' + '34')
1234
```

Q9.

```
>>> 'man' != 'men'
True
```

Q10.

```
>>> 3*'PYTHON'
'PYTHONPYTHONPYTHON'
```

Q11. `str = 'Welcome to Python'``print(str.isalnum())`

```
== RESIAI
False
```

Q12.

```
>>> 'Hello'.isalpha()
True
```

Q13.

```
>>> '14-10-2106'.isdigit()
False
```

Q14. `print('hello'.islower())`

```
-----
True
---
```

Q15.

```
type Copyright, or
>>> '\t'.isspace()
True
>>>
```

**Q16.** `str = 'Hello'`

`print (str. starts with('he'))`

```
==== RES1
False
^^^
```

**Q17.** `str = 'Hello, welcome to the world of Python'`

`print (str. find('o'))`

```
====
4
---
```

**Q18.** `str = 'Hello, welcome to the world of Python'`

`print (str. find('if'))`

```
==== RES1
-1
>>>
```

**Q19.** `str = 'Hello, welcome to the world of Python'`

`print (str. rfind('of'))`

```
==== K
28
>>>
```

**Q20.** `str = 'Hello, welcome to the world of Python'`

`print (str. count('o'))`

```
==== K
6
```

**Q21.**

```
>>> 'us' not in 'success'
True
```

**Q22.**

```
>>> 'mi' in 'ours'
False
```

**Q23.** `for i in 'Python':`

`print (2*i, end="")`

```
| PPyttthhoonn
```

**Q24.** `string='abcdabcdabcd'`  
`print (string. find ('cd', 3))`

```
| 6
|
|
```

**Q25.** `string='abcdabcdabcdabcdabcd'`  
`print (string. find ('cd', 7,13))`

```
====
| 10
|
|
```

**Q26.** `a = 10`  
`b = 20`  
`print ('3**4 = %d and %d * %d = %f' % (3**4, a, b, a * b))`

```
-----
| 3**4 = 81 and 10 * 20 = 200.000000
|
|
```

**Q27.** `print ('%d %f %s' % (7, 15, 28))`  
`print ('%-.2f' % 369)`  
`print ('%-10.2f%-10.2f' % (91, 23.456))`

```

| 7 15.000000 28
| 369.00
| 91.00      23.46
|
|
```

**Q28.** `str1 = 'Welcome!'`  
`str2 = 'to Python'`  
`str3 = str1[:2] + str2[len(str2)- 2:]`  
`print(str3)`

```

| Weon
|
|
```

**Q29.** `print ('She sells sea shells on the sea shore'. find ('sea', 3, -6))`

```
| 10
|
|
```

**Q30.** `str = 'Welcome to the world of Python'`  
`print (str [:10]. find('t'))`

```

| 8
|
|
```



**Q31.** `str = 'Welcome to the world of Python'`

`start = 3`

`end = 10`

`print (str [start: end])`

`come to`

**Q32.** `str = 'Hello'`

`print (str. lower (). starts with('h'))`

`True`

**Q33.**

`>>> 'In %d years I have saved %g %s.' % (3, 4.5, 'lakh rupees')`

`'In 3 years I have saved 4.5 lakh rupees.'`

**Q34.**

`>>> ', '.join(['Sun', 'Stars', 'Planets'])`

`'Sun, Stars, Planets'`

**Q35.**

`>>> ' '.join(['Welcome', 'to', 'the', 'world', 'of', 'Python!'])`

`'Welcome to the world of Python!'`

**Q36.**

`>>> 'Good morning students'.split()`  
`['Good', 'morning', 'students']`

**Q37.**

`>>> 'WelcomeHellotoHellotheHelloworldHelloofHelloPython!'.split('Hello')`  
`['Welcome', 'to', 'the', 'world', 'of', 'Python!']`

**Q38.**

`>>> print('One'+ 'Two'*2)`  
`OneTwoTwo`

**Q39.** `s = 'abcdefghijkl'`

```
print (s [-100: -5], s [-100:5])
```

```
abcdefghijkl  
| abcdefg abcde
```

**Q40.** `s1 = 'HELLO'`

```
s2 = s1 + s1[-1]
```

```
print(s2)
```

```
print (s1[: -1])
```

```
print(s1[-1:])
```

```
HELLOO  
HELL  
O
```

**Q41.** `str = 'xyz'`

```
while(len(str)<=4):
```

```
    if (str [-1] == 'z'):
```

```
        str = str [0:3] +'c'
```

```
    elif 'a' in str:
```

```
        str = str [0] + 'bb'
```

```
    elif 'x' not in str:
```

```
        str = '1'+str [1:] +'z'
```

```
    else:
```

```
        str = str+'*'
```

```
print(str)
```

```
xyzc*
```

**Q42.** `str = 'helloworldofpython'`

```
print (str [10], str [11:30])
```

```
o fpython
```

**Q43.** `str1 = "PYTHONPRO"`

```
str2 = "PYTHON\PRO"
```

```
print(len(str1)>len(str2))
```

```
False
```



**Q44.** `str='helloworldofpython'`  
`print (str [10], str [11:30])`  
`o fpython`

**Q45.**

```
>>> str1= 'PYTHON'; str1[-4]
'T'
```

**Q46.** `list = ['abc', 'def', 'ghi', 'jkl']`  
`print (list [1: -1])`  
`list [0:2] = 'xyz'`  
`['def', 'ghi']`

**Q47.** `list = ['abc', 'def', 'ghi', 'jkl', [1,2,3,4,5]]`  
`print (list [4][2])`  
`3`

**Q48.** `list = ['p','r','o','g','r','m','m','i','n','g']`  
`print (list [2:5])`  
`print (list [: -5])`  
`print (list [5:])`  
`print (list [:])`  
`['o', 'g', 'r']`  
`['p', 'r', 'o', 'g', 'r']`  
`['m', 'm', 'i', 'n', 'g']`  
`['p', 'r', 'o', 'g', 'r', 'm', 'm', 'i', 'n', 'g']`  
`>>>`

**Q49.** `even = [2,4,6]`  
`print (even + [10, 12, 14])`  
`print(even*2)`  
`even. insert (1,0)`  
`print(even)`  
`del even [2]`  
`print(even)`  
`[2, 4, 6, 10, 12, 14]`  
`[2, 4, 6, 2, 4, 6]`  
`[2, 0, 4, 6]`  
`[2, 0, 6]`

**Q50.** `list = ['p','r','o','g','r','a','m']`

```
list.remove('p')
```

```
print(list)
```

```
print (list.pop (1))
```

```
print(list)
```

```
print(list.pop())
```

```
print(list)
```

```
['r', 'o', 'g', 'r', 'a', 'm']
o
['r', 'g', 'r', 'a', 'm']
m
['r', 'g', 'r', 'a']
```

**Q51.** `list = [9,4,3,8,0,2,3,6]`

```
print (list.index (3))
```

```
print (list.count (8))
```

```
list.sort ()
```

```
print(list)
```

```
list.reverse ()
```

```
print(list)
```

```
print (0 in list)
```

```
2
1
[0, 2, 3, 3, 4, 6, 8, 9]
[9, 8, 6, 4, 3, 3, 2, 0]
True
```

**Q52.** `list = [2 ** x for x in range (5)]`

```
print(list)
```

```
[1, 2, 4, 8, 16]
```

**Q53.** `countries = ['India','Sri Lanka','New Zealand', 'Japan', 'Russia']`

```
for index, country in enumerate (countries):
```

```
print ('The country, ' + country + ', is at position ' + str(index) + ")
```

```
The country, India, is at position 0
The country, Sri Lanka, is at position 1
The country, New Zealand, is at position 2
The country, Japan, is at position 3
The country, Russia, is at position 4
```

**Q54.** `list = [(1, 2), [3, 4], '56', 78, 9.0]`

`print (list [0], type (list [0]))`

`print (list [2:3], type (list [0:1]))`

`print (list [2], type (list [2]))`

```
=====
(1, 2) <class 'tuple'>
['56'] <class 'list'>
56 <class 'str'>
```

**Q55.** `item = [x+y for x in 'cup' for y in 'pen']`

`print(item)`

```
['cp', 'ce', 'cn', 'up', 'ue', 'un', 'pp', 'pe', 'pn']
```

**Q56.** `print ([x+y for x in 'cup' for y in 'pen' if x!= 't' and y!= 'o'])`

```
=====
['cp', 'ce', 'cn', 'up', 'ue', 'un', 'pp', 'pe', 'pn']
```

**Q57.** `list = [ [1,2] *3] *4`

`print(list)`

```
==== RESTART: C:/Users/aksha/AppData/Local/Programs/Python/Python36/jd.py ====
[[1, 2, 1, 2, 1, 2], [1, 2, 1, 2, 1, 2], [1, 2, 1, 2, 1, 2], [1, 2, 1, 2, 1, 2]]
```

**Q58.** `list = [10, 20, 30, 40, 50, 60, 70, 80, 90]`

`print (list [-4: -1])`

`print (list [-1: -4])`

`print (list [-5:])`

`print (list [-6: -2:2])`

`print (list [: -1])`

```
=====
[60, 70, 80]
[]
[50, 60, 70, 80, 90]
[40, 60]
[90, 80, 70, 60, 50, 40, 30, 20, 10]
```

**Q59.** `list = [[10, 20, [30, 40, [50, 60]]]]`

`print (list [0])`

`print (list [0][2])`

`print (list [0][2][2])`

`print (list [0][0])`

```

print (list [0][2][1])
print (list [0][2][2][0])
[10, 20, [30, 40, [50, 60]]]
[30, 40, [50, 60]]
[50, 60]
10
40
50

```

**Q60.** List = [100, 90, 80, 70, 60, 50]

```
List [2] = List [1] - 20
```

```
if 30 in List:
```

```
    print (List [3])
```

```
else:
```

```
    print (List [4])
```

```
60
```

**Q61.** List = list (range (2, 20, 3))

```
print (List [5])
```

```
17
```

**Q62.** List = [-5, -3, 0, 3, 6]

```
print ([x*2 for x in List])
```

```
print ([x for x in List if x >= 0])
```

```
[-10, -6, 0, 6, 12]
[0, 3, 6]
```

**Q63.** print ((x, x\*2) for x in range (5))

```
[(0, 0), (1, 2), (2, 4), (3, 6), (4, 8)]
```

**Q64.** List = [[1,2,3], [4,5,6], [7,8,9]]

```
print ([val for x in List for val in x])
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

**Q65.** DC = [-100, 0, 32, 40, 100]

```
DF = map (lambda temp: (9.0/5) *temp + 32, DC)
```

```
print (DF)
```

```
<map object at 0x000000143E0AAC128>
```

**Q66.** `List = [-10,20, -30,40, -50]`  
`if all([abs(i)<30 for i in List]):`  
`print('Hi')`  
`else:`  
`print('Bye')`

`Bye`

**Q67.** `def add_two(x):`  
`return x+2`  
`List = [10,20,30,40,50]`  
`result = list (map (add_two, List))`  
`print(result)`

`[12, 22, 32, 42, 52]`

**Q68.** `List = [13,26,39,52,64]`  
`print (list (filter (lambda x:x%2==1, List)))`  
`[13, 39]`

**Q69.** `str = 'abcdefghijklmno'`  
`for i in range (0, len(str), 2):`  
`print(str[i], end = ' ')`

`a c e g i k m o`

**Q70.** `l1 = [1,2,3]`  
`l2 = ['a','b','c']`  
`l3 = [1.2,3.4,5.6]`  
`l4 = l1 + l2 + l3`  
`print(l4)`

`[1, 2, 3, 'a', 'b', 'c', 1.2, 3.4, 5.6]`

**Q71.** `print ([1,2,8,9] < [1,2,8,9,10])`  
`True`

**Q72.** `l = [1,2,3,4,5,6,7,8,9,10]`

```
print(l[-1])
print(l[l[0]])
print(l[l[-8]])
print (l [l [l [0] +1] +2)

10
2
4
6
```

**Q73.** `msg = ['PYTHON','is','a', ['simple','interpreted','OOP'],'language']`

```
print (msg [2:4])
print (msg [2:4][1][2])
print (msg [2:4][1][2][1])
print ('im' in msg [2:4][1][2][1])
print (msg [2:4][1][1] [3:])
print (msg [1] +msg [4])

['a', ['simple', 'interpreted', 'OOP']]
OOP
O
False
rpreted
islanguage
```

**Q74.**

```
>>> [1,2,3] +[1,2,3] == [1,2,3]*2
True
```

**Q75.** `msg = ['PYTHON','is','a', ['simple','interpreted','OOP'],'language']`

```
print (msg [:2])
print ('n' in msg [4])
print (msg [2] in msg [4])

['PYTHON', 'a', 'language']
True
True
```

**Q76.** `l = [1,2,3]`

```
print ((l + [4,5,6]) [3])
```



**Q77.**

```
>>> [1,2] == [1,2]
True
```

**Q78.**

```
>>> [1,2] is [1,2]
False
```

**Q79.** `L = [1,2,3,4,5,6,7,8,9,10,11,12,13,14]`

```
print(L[: -1])
print(L[-1: -2: -3])

[14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
[14]
```

**Q80.** `for i in [1,2,3]:``for j in [4,5,6]:``print(i, j, end = '')`

```
1 4 1 5 1 6 2 4 2 5 2 6 3 4 3 5 3 6
```

**Q81.** `count = 0``for i in range(5):``for j in range(10):``count = count + 1``print(count)`

O/P: 1 to 50 printed

**Q82.** `import random``city_list = ['New York', 'Los Angeles', 'Chicago', 'Houston', 'Philadelphia']``print('Select random element from list - ', random.choice(city_list))`

```
Select random element from list - New York
```

**Q83.** `tup1 = (1,2,3)``tup2 = (1.0,2.0,3.0)``print(tup1 == tup2)`

```
True
```

**Q84.** `Tup = ('abc', 'def')`

```
(key, value) = Tup
```

```
print (key, value)
```

```
| abc def
```

**Q85.** `Tup = (1,2,3)`

```
Add_Tup = Tup + Tup
```

```
print (Add_Tup)
```

```
Mul_Tup = Tup * 3
```

```
print (Mul_Tup)
```

```
|-----|
(1, 2, 3, 1, 2, 3)
```

```
(1, 2, 3, 1, 2, 3, 1, 2, 3)
```

**Q86.** `msg = 'HelloWorld'`

```
pairs = []
```

```
for i in range (1, len(msg), 2):
```

```
    first = msg [i - 1]
```

```
    second = msg[i]
```

```
    pairs.append ((first, second))
```

```
for item in pairs:
```

```
    print(item)
```

```
| ('H', 'e')
```

```
| ('l', 'l')
```

```
| ('o', 'W')
```

```
| ('o', 'r')
```

```
| ('l', 'd')
```

**Q87.** `Tup = (1, 'abc')`

```
List = [1, 'abc']
```

```
print (Tup == List)
```

```
print (Tup == tuple (List))
```

```
print (list (Tup) == List)
```

```
print ((1, 2) + (3, 4))
```

```
|-----|
False
```

```
True
```

```
True
```

```
(1, 2, 3, 4)
```

**Q88.** list = ['Good', 'Morning']

```
y, x = list
```

```
print (x, y)
```

```
----- RESTART: C:/Users/ANSHU/A
Morning Good
```

**Q89.** A = ('Chinu', 30, 'Female')

```
B = ('Varun', 32, 'Male')
```

```
for i in [A, B]:
```

```
    print ('%s is a %d year old %s' %i)
```

```
----- RESTART: C:/Users/ANSHU/A
Chinu is a 30 year old Female
Varun is a 32 year old Male
```

**Q90.** Tup = ('Good')

```
for i in range (4):
```

```
    Tup = (Tup)
```

```
    print (Tup)
```

```
-----
Good
Good
Good
Good
```

**Q91.** Tup1='a','bcd',12.34

```
Tup2=Tup1, (5,6,7,8)
```

```
print (Tup2)
```

```
-----
(('a', 'bcd', 12.34), (5, 6, 7, 8))
```

**Q92.** Tup = (1, 2, [3, 4])

```
Tup [2][0] = 5
```

```
print (Tup)
```

```
-----
(1, 2, [5, 4])
```

**Q93.** Tup = ('Good Morning')

```
print (Tup.index('M'), end = ' ')
```

```
print (Tup.index('n', 5))
```

```
print (Tup.index('r',4,8))
```

```
-----
5 8
7
```

**Q94.** `tup = (1,2,3,4,5,6,7,8,9)`

```
print (tup [3:5] *3)
```

```
| (4, 5, 4, 5, 4, 5)
```

**Q95.** `tup = (10,7,8,9,6,3,4,5,0,1,2)`

```
print (tup [0])
```

```
print (tup [tup [3]])
```

```
print (tup [-4])
```

```
print (tup [tup [-2]])
```

```
print (tup [tup [tup [7]]-4])
```

```
print (tup [10*2-15+3])
```

```
| 10  
| 1  
| 5  
| 7  
| 2  
| 0
```

**Q96.** `t = (1,2,3)`

```
a, b, c = (x, y, z) = t
```

```
print (a, b, c)
```

```
print (x, y, z)
```

```
==== RES  
| 1 2 3  
| 1 2 3
```

**Q97.** `t=('a')`

```
print(type(t))
```

```
t=('a')
```

```
print(type(t))
```

```
| <class 'str'>  
| <class 'str'>
```

**Q98.** `t1 = (1,2,3)`

```
t2 = (4,5)
```

```
print (t1 + t2)
```

```
t1 = (1,2,3)
```

```
t2 = t1 * (3)
```

```
print(t2)
```

```
(1, 2, 3, 4, 5)
```

```
(1, 2, 3, 1, 2, 3, 1, 2, 3)
```



**Q99.** t1 = (1,2,3)

t2 = ('1','2')

print (t1 + t2)

```
==== RESTART: C:/Use
(1, 2, 3, '1', '2')
```

**Q100.** t = (78,98,80,76,53,54,78,98,87,74)

print (t [3:])

print (t [:6])

print (t [-4:])

print (t [-4:4])

print (t [:])

```
(76, 53, 54, 78, 98, 87, 74)
(78, 98, 80, 76, 53, 54)
(78, 98, 87, 74)
()
(78, 98, 80, 76, 53, 54, 78, 98, 87, 74)
```

**Q101.** t=('Python','Programming','is','fun')

(a, b, c, d) = t

print (t [0][0] + t [1][1] +t [1])

```
PrProgramming
```

**Q102.** t1 = 'a','b'

t2 = ('a','b')

print(t1==t2)

```
True
```

**Q103.** t = ('a', ('b', ('c', ('d',))))

print(len(t))

print (t [1][0])

print ('c' in t)

```
2
b
False
```

**Q104.** `t = (1,2, (3,4),5, (6, (7,8),9))`

```
print(len(t))
print (t [3] +10)
print (t [t [1]])
print (t [t [1]][1] *10)
```

```
5
15
(3, 4)
40
```

**Q105.** `t = ((1,2),) *7`

```
print(t)
print (len (t [3:8]))
```

```
----- RESTART: C:/Users/akshu/AppData/Local/Programs/Python/P
((1, 2), (1, 2), (1, 2), (1, 2), (1, 2), (1, 2), (1, 2))
4
\\
```

**Q106.** `tup = ('abc','def')`

```
x, y = tup
print (x, y)
```

```
abc def
```

**Q107.** `students = ('Bhavya', 'Era', 'Falguni', 'Huma')`

```
index = students.index('Falguni')
print ('Falguni is present at location: ', index)
index = students.index('Isha')
print ('Isha is present at location: ', index)
```

```
----- RESTART: C:/Users/akshu/AppData/L
Falguni is present at loca tion : 2
```

**Q108.** `Dict = {'India': 'New Delhi', 'Nepal' : 'Kathmandu','USA':'Washington DC'}`

```
del Dict['Nepal']
for key, val in Dict.items():
    print (key, val)
```

```
-----
India New Delhi
USA Washington DC
```

**Q109.** Dict = {'India': 'New Delhi', 'Nepal': 'Kathmandu', 'USA': 'Washington DC'}

```
print (Dict.get('Russia'))
print (Dict.get ('Pakistan', 'No Idea'))

None
No Idea
```

**Q110.** Studs = {'Mitanshi', 'Harshita', 'Pritika'}

```
Toppers = {}. fromkeys (Studs, 0)
print (Toppers)
Toppers['Mitanshi'] = 97
Toppers['Harshita'] = 92
Toppers['Pritika'] = 89
Toppers.setdefault('Nisha', -1)
print (Toppers)

{'Harshita': 0, 'Mitanshi': 0, 'Pritika': 0}
{'Harshita': 92, 'Mitanshi': 97, 'Pritika': 89, 'Nisha': -1}
```

**Q111.** Toppers = {}

```
Toppers['Mitanshi'] = 97
Toppers['Harshita'] = 92
Toppers['Pritika'] = 89
print ('Harshita got')
str (Toppers.get('Harshita')) + 'marks'

Harshita got
```

**Q112.** rec = {'Name': {'First': 'Chaitanya', 'Last': 'Raj'}, 'Marks': [80, 76, 84], 'Course': 'BTech'}

```
print(rec['Name'])
print(rec['Name'] ['Last'])
print(rec['Marks'])
rec['Marks']. append (72)
print(rec)

{'First': 'Chaitanya', 'Last': 'Raj'}
Raj
[80, 76, 84]
{'Name': {'First': 'Chaitanya', 'Last': 'Raj'}, 'Marks': [80, 76, 84, 72], 'Course': 'BTech'}
```

**Q113.** Dict = {'Amna':4,'Brij':2,'Chaitanya':5,'Divyanka':3}

```
s = 0
for v in Dict.values():
    s = s + v
print(s)
14
```

**Q114.** Dict = {'Amna':4,'Brij':2,'Chaitanya':5,'Divyanka':3}

```
n = 'Chaitanya'
v = 10
if n in Dict:
    Dict[n] = v
print (Dict)
{'Amna': 4, 'Brij': 2, 'Chaitanya': 10, 'Divyanka': 3}
```

**Q115.** Dict = {'a':1, 'b':2, ('c','d'):3}

```
print (Dict)
{'a': 1, 'b': 2, ('c', 'd'): 3}
{'a': 1, 'b': 2, ('c', 'd'): 3}
```

**Q116.** Dict = {'a':1, 'b':2, 'c':3, 'd':4}

```
v = Dict['c']
if v in Dict:
    print('FOUND')
else:
    print ('NOT FOUND')
=== RESTART:
NOT FOUND
```

**Q117.** Dict = {'a':1, 'b':2, 'c':3, 'd':4}

```
k = ''
for i in Dict:
    if i > k:
        k = i
v = Dict[i]
```



```
print (k, v)
List = list (Dict.items())
List. sort ()
print (List)
```

```
d 4
[('a', 1), ('b', 2), ('c', 3), ('d', 4)]
```

**Q118.** Dict = {'a':1, 'b':2, 'c':3, 'd':4, 'e': [5,6,7,8]}

```
print (Dict)
```

```
{'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': [5, 6, 7, 8]}
```

**Q119.** Dict = {10:'TEN', 'HUNDRED':100, (1,2) :('ONE', 'TWO')}

```
print (Dict)
```

```
{10: 'TEN', 'HUNDRED': 100, (1, 2): ('ONE', 'TWO')}
```

**Q220.** List = [10,10,10,20,30,30,20,40,40,40]

```
counts= {}
```

```
L= []
```

```
for i in List:
```

```
    if i not in L:
```

```
        L.append(i)
```

```
        counts[i] = 1
```

```
    else:
```

```
        counts[i] += counts[i]
```

```
print(counts)
```

```
{10: 4, 20: 2, 30: 2, 40: 4}
```

**Q221.** Dict = {}

```
Dict [1] = 1
```

```
Dict ['1'] = 2
```

```
Dict [1] += 1
```

```
sum = 0
```

```
for i in Dict:
```

```
    sum += Dict[i]
```

```
print(sum)
```

```
4
```

## CHAPTER 3: Functions and Modules

### GIVE OUTPUT

Q1.

```
num = 10
def show():var = 20
print("In Function var is - ", num)
show()
print("Outside function, var is -", num)
```

OUTPUT

```
In Function var is - 10
Outside function, var is - 10
```

Q2.

```
def f():
    s = "Hello World!"
    print(s)
s = "Welcome to Python Programming"
f()
print(s)
```

OUTPUT

```
Hello World!
Welcome to Python Programming
```

Q3.

```
def f():
    global var
    print(var)
    var = 10
    print(var)
var = 100
f()
```

OUTPUT

100

10

**Q4.**

```
def display (str):print(str+"!")  
display ("Hello World")
```

OUTPUT

Hello World!

**Q5.**

```
def sqr(x):print(x*x)
```

```
sqr(10)
```

OUTPUT

100

**Q6.**

```
def mul_twice(x,y):print(x*y)
```

```
mul_twice(5, 10)
```

OUTPUT

50

**Q7.**

```
def func():
```

```
    global x
```

```
    print("x =", x)
```

```
    x = 100
```

```
    print('x is now = ', x)
```

```
x = 10
```

```
func()
```

```
print('x =', x)
```

OUTPUT

x = 10

x is now = 100

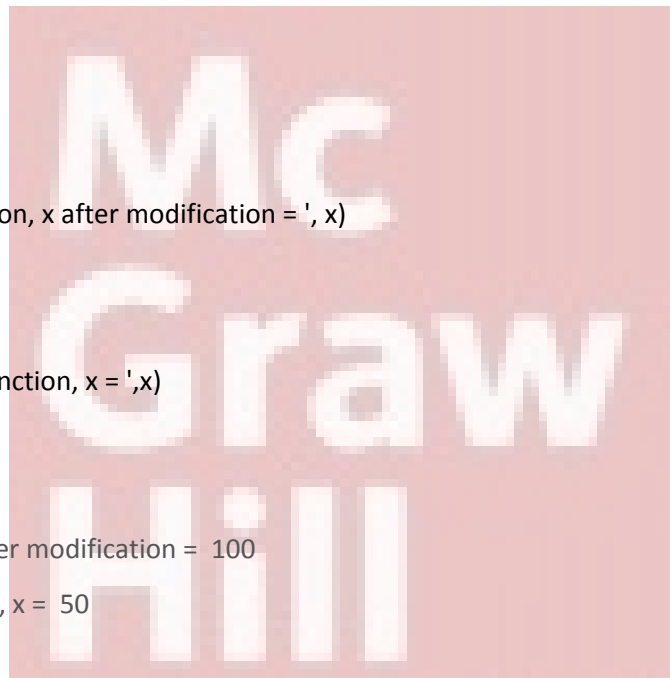
x = 100

**Q8.**

```
def func1():  
    var = 3  
    func2(var)  
def func2(var):  
    print(var)  
func1()  
OUTPUT  
3
```

**Q9.**

```
def func(x):  
    print('x = ', x)  
    x = 100  
    print('In Function, x after modification = ', x)  
x = 50  
func(x)  
print('Outside Function, x = ',x)  
OUTPUT  
x = 50  
In Function, x after modification = 100  
Outside Function, x = 50
```

**Q10.**

```
def display( str ):  
    print(str)  
    return;  
display('Hello World !!')  
display("Welcome to Python Programming")  
OUTPUT  
Hello World !!  
Welcome to Python Programming
```

**Q11.**

```
def sum( num1, num2 ):
    total = num1 + num2
    print('Inside function, Total= ', total)
    return total;

total = sum( 10, 20 )
print('Outside the function,Total = ', total)
```

OUTPUT

Inside function, Total= 30

Outside the function,Total = 30

**Q12.**

```
def min(x,y):
    if x<y:
        return x
    else:
        return y
print(min(4, 7))
```

OUTPUT

4

**Q13.**

```
def add(x, y):
    sum = x + y
    return sum

print('This won't be printed')
print(add(10,20))
```

OUTPUT

This won't be printed

30



**Q14.**

```
def say(message, repeat_it = 2):  
    print(message * repeat_it)  
say('Hello')  
say('Hello', 5)  
  
OUTPUT  
HelloHello  
HelloHelloHelloHelloHello
```

**Q15.**

```
def func(x, y = 100, z = 1000):  
    print('x = ', x, 'y = ', y, 'and z = ', z)  
func(5, 15, 25)  
func(35, z = 55)  
func(y = 70, x = 200)  
  
OUTPUT  
x = 5 y = 15 and z = 25  
x = 35 y = 100 and z = 55  
x = 200 y = 70 and z = 1000
```

**Q16.**

```
def greet(*names):  
    for name in names:  
        print('Hello',name)  
greet('Aryan','Nikita','Cahitanya')  
  
OUTPUT  
Hello Aryan  
Hello Nikita  
Hello Cahitanya
```

**Q17.**

```
def func( arg1, *var):  
    "This prints arbitrary arguments"
```

```

print(arg1)

for i in var:
    print(i)
    return;

func("Score is : ", 10, 20, 30)
func( "\n Average Score = ", 20)

```

OUTPUT

Score is :

10

Average Score =

20

**Q18.**

```

expo_3 = lambda x: x ** 3
print(expo_3(5))

```

OUTPUT

125

**Q19.**

```

add_five = lambda n: n + 5
mult_add_five = lambda n: add_five(n * 10)
print(mult_add_five(9))

```

OUTPUT

95

**Q20.**

```

def func():
    """Do nothing.
    Nothing doing.
    """
    pass
print(func.__doc__)

```

OUTPUT

Do nothing.

Nothing doing.

**Q21.**

```
def C_to_F(c):  
    return c * 9/5 + 32  
print(C_to_F(37))
```

OUTPUT

98.6

**Q22.**

```
def pow(x, y=3):  
    r = 1  
    for i in range(y):  
        r = r * x  
    return r  
print(pow(5))  
print(pow(2, 5))
```

OUTPUT

5

2



**Q23.**

```
def display(name, deptt, sal):  
    print("Name: ", name)  
    print("Department: ", deptt)  
    print("Salary: ", sal)  
display(sal = 100000, name="Tavisha", deptt = "IT")  
display(deptt = "HR", name="Dev", sal = 50000)
```

OUTPUT

Name: Tavisha

Department: IT



Salary: 100000

Name: Dev

Department: HR

Salary: 50000

**Q24.**

```
def display(mesg):  
    return mesg + "!"  
print_str = display  
str = print_str("Hello")  
print(str)
```

OUTPUT

Hello!

**Q25.**

```
from random import randint as r  
for i in range(10):  
    value = r(1,100)  
print(value)
```

OUTPUT

9

**Q26.**

```
def is_even(x):  
    if x==0:  
        return True  
    else:  
        return is_odd(x-1)  
def is_odd(x):  
    return not is_even(x)  
print(is_even(22))
```

OUTPUT

True



**Q27.**

```
def display(x):  
    for i in range(x):  
        print(i)  
    return
```

```
display(5)
```

OUTPUT

0



## CHAPTER 4: Data Handling Using Numpy

**Q1.**

```
import numpy as np
a = np.arange(5, 14, 2)
print(a)
```

OUTPUT

```
[ 5  7  9 11 13]
```

**Q2.**

```
import numpy as np
a = np.linspace(5, 25, 4)
print(a)
```

OUTPUT

```
[ 5.    11.66666667 18.33333333 25.    ]
```

**Q3.**

```
import numpy as np
x = np.arange(10)
x=np.array([0, 1, 2, 3, 4, 5, 6, 7,8, 9])
print(x[:5])
print(x[5:])
print(x[4:7])
print(x[::2])
print(x[1::2])
```

OUTPUT

```
[0 1 2 3 4]
```

```
[5 6 7 8 9]
```

```
[4 5 6]
```

```
[0 2 4 6 8]
```

```
[1 3 5 7 9]
```

**Q4.**

```
import numpy as np
x = np.array([1, 2, 3])
```

Copyright © 2023 by McGraw Hill Education (India) Private Limited

```
print(x.reshape(1, 3))
print(x[np.newaxis, :])
print(x.reshape(3, 1))
print(x[:, np.newaxis])
```

OUTPUT

```
[[1 2 3]]
```

```
[[1]]
```

```
[2]
```

```
[3]]
```

```
[[1 2 3]]
```

```
[[1]]
```

```
[2]
```

```
[3]]
```

**Q5.**

```
import numpy as np
grid = np.array([[1, 2, 3],[4, 5, 6]])
a=np.concatenate([grid, grid])
b=np.concatenate([grid, grid],axis=1)
print(a)
print(b)
```

OUTPUT

```
[[1 2 3]
```

```
[4 5 6]
```

```
[1 2 3]
```

```
[4 5 6]]
```

```
[[1 2 3 1 2 3]
```

```
[4 5 6 4 5 6]]
```

**Q6.**

```
import numpy as np
x = np.array([1, 2, 3])
grid = np.array([[9, 8, 7],[6, 5, 4]])
np.vstack([x, grid])
y = np.array([[99], [99]])
```

```
np.hstack([grid, y])
```

OUTPUT

```
[[1 2 3]
 [9 8 7]
 [6 5 4]]
[[ 9 8 7 99]
 [ 6 5 4 99]]
```

**Q7.**

```
import numpy as np
x = [1, 2, 3, 99, 99, 3, 2, 1]
x1, x2, x3 = np.split(x, [3,5])
print(x1, x2, x3)
```

OUTPUT

```
[1 2 3] [99 99] [3 2 1]
```

**Q8.**

```
import numpy as np
x = np.ones((4,3))
print("Dimensions of x : ",x.shape)
y = np.random.random((1,3))
print("Dimensions of y : ",y.shape)
print(x + y)
```

OUTPUT

```
Dimensions of x : (4, 3)
Dimensions of y : (1, 3)
[[1.95856933 1.62842552 1.59046076]
 [1.95856933 1.62842552 1.59046076]
 [1.95856933 1.62842552 1.59046076]
 [1.95856933 1.62842552 1.59046076]]
```

**Q9.**

```
import numpy as np
arr = np.array([[1,2,3,4],[5,6,7,8],[9,0,1,2],[3,4,5,6]])
print("Elements at (1,0),(0,1), (1,2) and (0,0) \n")
print(arr[[1, 0, 1, 0],[0, 1, 2, 0]])
```

Copyright © 2023 by McGraw Hill Education (India) Private Limited

OUTPUT

Elements at (1,0),(0,1), (1,2) and (0,0)

[5 2 7 1]

**Q10.**

```
import numpy
a = numpy.array([1, 2, 3, 4,5])
b = numpy.array([10, 20, 30,40, 50])
newArray = numpy.append(a, b)
print("The new array = ",newArray)
```

OUTPUT

The new array = [ 1 2 3 4 5 10 20 30 40 50]

**Q11.**

```
import numpy as np
a =np.array([[1,2],[3,4],[5,6]])
print( np.insert(a,3,[11,12]))
print( np.insert(a,1,[11],axis= 0) )
print( np.insert(a,1,11,axis =1))
```

OUTPUT

[ 1 2 3 11 12 4 5 6]

[[ 1 2]

[11 11]

[ 3 4]

[ 5 6]]

[[ 1 11 2]

[ 3 11 4]

[ 5 11 6]]

**Q12.**

```
import numpy as np
arr = np.arange(6).reshape(2,3)
print("1D arr : \n", arr)
print("Shape : ", arr.shape)
a = np.insert(arr, (2, 4), 9)
print("\nInsertion at twopoints : ", a)
```

Copyright © 2023 by McGraw Hill Education (India) Private Limited

```
print("Shape : ", a.shape)
```

OUTPUT

1D arr :

```
[[0 1 2]
```

```
[3 4 5]]
```

Shape : (2, 3)

Insertion at twopoints : [0 1 9 2 3 9 4 5]

Shape : (8,)

### Q13.

```
import numpy as np
```

```
arr = np.arange(12).reshape(3,4)
```

```
print("\n\n2D arr : \n", arr)
```

```
print("Shape : ", arr.shape)
```

```
a = np.insert(arr, (0, 3), 66,axis = 1)
```

```
print("\nInsertion at two points : \n", a)
```

```
print("Shape : ", a.shape)
```

OUTPUT

2D arr :

```
[[ 0  1  2  3]
```

```
[ 4  5  6  7]
```

```
[ 8  9 10 11]]
```

Shape : (3, 4)

Insertion at two points :

```
[[66 0 1 2 66 3]
```

```
[66 4 5 6 66 7]
```

```
[66 8 9 10 66 11]]
```

Shape : (3, 6)

### Q14.

```
import numpy as np
```

```
a = np.arange(12).reshape(3,4)
```

```
print(a)
```

```
print(np.delete(a,5) )
```

```
print(np.delete(a,1,axis = 1))
```

OUTPUT

Copyright © 2023 by McGraw Hill Education (India) Private Limited

```
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
[ 0  1  2  3  4  6  7  8  9 10 11]
[[ 0  2  3]
 [ 4  6  7]
 [ 8 10 11]]
```

**Q15.**

```
import numpy as np
a = np.array([1,2,3,4,5,6,7,8,9,10])
print(np.delete(a,np.s_[:2]))
```

OUTPUT

```
[ 2  4  6  8 10]
```

**Q16.**

```
import numpy as np
arr = np.arange(12).reshape(3,4)
print("arr : \n", arr)
print("Shape : ", arr.shape)
```

OUTPUT

arr :

```
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
```

Shape : (3, 4)

**Q17.**

```
import numpy
a = numpy.array([1, 2, 3])
newArray = numpy.delete(a, 1,axis = 0)
print(newArray)
```

OUTPUT

```
[1 3]
```



**Q18.**

```
import numpy
a = numpy.array([[1, 2, 3],[4, 5, 6], [10, 20, 30]])
newArray = numpy.delete(a, 1,axis = 0)
print(newArray)
```

OUTPUT

```
[[ 1  2  3]
 [10 20 30]]
```

**Q19.**

```
import numpy as np
arr = np.array([])
if(arr.size == 0):print("The given Array is empty")
else:print("The array = ", arr)
```

OUTPUT

The given Array is empty

**Q20.**

```
import numpy
a = numpy.array([1, 2, 3, 4,5, 6, 7, 8])
print("A subset of array a =", a[2:5])
```

OUTPUT

A subset of array a = [3 4 5]

**Q21.**

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8,9])
newarr = arr.reshape(3, 3)
print(newarr)
```

OUTPUT

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

**Q22.**

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5,6, 7, 8])
newarr = arr.reshape(2, 2, -1)
```

Copyright © 2023 by McGraw Hill Education (India) Private Limited

```
print(newarr)
```

OUTPUT

```
[[[1 2]
 [3 4]]
 [[5 6]
 [7 8]]]
```

**Q23.**

```
import numpy as np
```

```
arr = np.array([[1, 2, 3],[4, 5, 6]])
```

```
newarr = arr.reshape(-1)
```

```
print(newarr)
```

OUTPUT

```
[1 2 3 4 5 6]
```

**Q24.**

```
import numpy as np
```

```
a = np.arange(10, 1, -2)
```

```
newarr = a[np.array([3, 1, 2])]
```

```
print("\n Elements at these indices are:\n",newarr)
```

OUTPUT

```
Elements at these indices are:
```

```
[4 8 6]
```

**Q25.**

```
import numpy as np
```

```
x = np.array([1, 2, 3, 4, 5,6, 7, 8, 9])
```

```
arr = x[np.array([1, 3, -3])]
```

```
print("\n Elements are :\n",arr)
```

OUTPUT

```
Elements are:
```

```
[2 4 7]
```

**Q26.**

```
import numpy as np
```

```
a = np.arange(20)
```

Copyright © 2023 by McGraw Hill Education (India) Private Limited

```
print("\n a[-8:17:1] =",a[-8:17:1])
```

```
print("\n a[10:] = ",a[10:])
```

OUTPUT

```
a[-8:17:1] = [12 13 14 15 16]
```

```
a[10:] = [10 11 12 13 14 15 16 17 18 19]
```

**Q27.**

```
import numpy as np
```

```
b = np.array([[[1, 2, 3],[4,5, 6]],[[7, 8, 9],[10, 11, 12]]])
```

```
print(b[...,1]) #Equivalent to b[:, :, 1 ]
```

OUTPUT

```
[[ 2  5]
```

```
 [ 8 11]]
```

**Q28.**

```
import numpy as np
```

```
a = np.array([10, 40, 80, 50,100])
```

```
print(a[a>50])
```

OUTPUT

```
[ 80 100]
```

**Q29.**

```
import numpy as np
```

```
a = np.array([[1, 2], [3, 4]])
```

```
print(a)
```

OUTPUT

```
[[1 2]
```

```
 [3 4]]
```

**Q30.**

```
import numpy as np
```

```
a = np.array([1, 2, 3,4,5],ndmin = 2)
```

```
print(a)
```

OUTPUT

```
[[1 2 3 4 5]]
```

**Q31.**

```
import numpy as np  
a = np.array([1, 2, 3], dtype= complex)  
print(a)
```

OUTPUT

```
[1.+0.j 2.+0.j 3.+0.j]
```



## CHAPTER 5: Python Pandas

**Q1.**

```
import pandas as pd
import numpy as np

unsorted_df=pd.DataFrame(np.random.randn(10,2),index=[1,4,6,2,3,5,9,8,0,7],columns =
['col2','col1'])

sorted_df=unsorted_df.sort_index()

print(sorted_df)
```

OUTPUT

	col2	col1
0	0.277258	1.008216
1	-0.938570	0.521794
2	-0.133053	-0.717795
3	-0.195689	-0.196641
4	-2.626200	-0.246940
5	-0.502415	1.229395
6	-1.090738	0.387574
7	-0.528015	-3.326877
8	1.267449	-1.180548
9	0.098834	0.877198

**Q2.**

```
import pandas as pd
import numpy as np
import math

# user defined function
def scale(num):return(num*10)

df = pd.Series([1,2,3,4,5,6,7,8,9,10])

df = df.pipe(scale)

print(df)
```

OUTPUT

0 10

1 20

2 30

3 40

4 50

5 60

6 70

7 80

8 90

9 100

dtype: int64

**Q3.**

```
import pandas as pd
```

```
import numpy as np
```

```
import math
```

```
# user defined function
```

```
def scale(num):return(num*10)
```

```
df=pd.DataFrame(6*np.random.randn(6,3),columns=['A','B','C'])
```

```
df = df.apply(scale,axis=1)
```

```
print(df)
```

OUTPUT

	A	B	C
0	37.158775	-38.698028	9.468330
1	-65.031939	-36.950604	82.803846
2	13.082511	90.508916	138.229122
3	-39.713790	-10.848235	51.794373
4	126.520031	20.604768	-56.974467
5	-18.470645	28.313269	80.901387

**Q4.**

```
import pandas as pd
```

```
import numpy as np
```

Copyright © 2023 by McGraw Hill Education (India) Private Limited

```
def adder(ele1,ele2):return ele1+ele2

df=pd.DataFrame(np.random.randn(5,3),columns=['col1','col2','col3'])
df.pipe(adder,2)
print(df.apply(np.mean))

OUTPUT

col1  0.068159
col2  -0.584054
col3   0.325352
dtype: float64
```

**Q5.**

```
import pandas as pd
import numpy as np
df=pd.DataFrame(np.random.randn(5,3),columns=['col1','col2','col3'])
df.apply(np.mean)
print(df.apply(np.mean))

OUTPUT

col1  0.766531
col2  -0.597091
col3   0.076810
dtype: float64
```

**Q6.**

```
import pandas as pd

data = {"Team": ["Red Sox", "RedSox", "Red Sox", "Red Sox", "RedSox", "Red Sox",
"Yankees", "Yankees", "Yankees", "Yankees", "Yankees", "Yankees"], "Pos": ["Pitcher",
"Pitcher", "Pitcher", "Not Pitcher", "NoPitcher", "Not Pitcher", "Pitcher", "Pitcher", "Pitcher",
"NotPitcher", "Not Pitcher", "NotPitcher"],
"Age": [24, 28, 40, 22, 29, 33, 31, 26, 21, 36, 25, 31]}

df = pd.DataFrame(data)

grouped_single = df.groupby('Team').agg({'Age': ['mean', 'min', 'max']})
print(grouped_single)

grouped_single.columns = ['age_mean', 'age_min', 'age_max']
grouped_single = grouped_single.reset_index()
print(grouped_single)
```

```
grouped_multiple = df.groupby(['Team', 'Pos']).agg({'Age': ['mean', 'min', 'max']})
grouped_multiple.columns = ['age_mean', 'age_min', 'age_max']
grouped_multiple = grouped_multiple.reset_index()
print(grouped_multiple)
```

OUTPUT

```
      Age
      mean min max
Team
Red Sox 29.750000 22 40
Red Sox 28.500000 28 29
Yankees 28.333333 21 36
```

```
      Team age_mean age_min age_max
0 Red Sox 29.750000    22    40
1 Red Sox 28.500000    28    29
2 Yankees 28.333333    21    36

      Team   Pos age_mean age_min age_max
0 Red Sox Not Pitcher    27.5    22    33
1 Red Sox   Pitcher    32.0    24    40
2 Red Sox NoPitcher    29.0    29    29
3 Red Sox   Pitcher    28.0    28    28
4 Yankees Not Pitcher    25.0    25    25
5 Yankees NotPitcher    33.5    31    36
6 Yankees   Pitcher    26.0    21    31
```

**Q7.**

```
import pandas as pd
import numpy as np

df = pd.DataFrame([[1, 2, 3],[4, 5, 6],[7, 8, 9],[np.nan, np.nan, np.nan]],columns=['A', 'B', 'C'])
print(df)
print(df.agg(['sum', 'min']))
print(df.agg({'A': ['sum', 'min'],'B': ['min', 'max']}))
print(df.agg("mean",axis="columns"))
print(df.agg("mean", axis="rows"))
```



OUTPUT

```

    A    B    C
0  1.0  2.0  3.0
1  4.0  5.0  6.0
2  7.0  8.0  9.0
3  NaN  NaN  NaN

    A    B    C
sum 12.0 15.0 18.0
min  1.0  2.0  3.0

    A    B
sum 12.0 NaN
min  1.0  2.0
max  NaN  8.0
0    2.0
1    5.0
2    8.0
3    NaN
dtype: float64
A    4.0
B    5.0
C    6.0
dtype: float64

```



**Q8.**

```

import pandas as pd
import numpy as np
column=['a','b','c','d','e']
index=['A','B','C','D','E']
df1 = pd.DataFrame(np.random.rand(5,5),columns=column,index=index)
column=['e','a','b','c','d']
print(df1.reindex(column,axis='columns'))

```

OUTPUT

```

    e      a      b      c      d
A  0.825389 0.341840 0.823577 0.669509 0.561256

```

```

B 0.054524 0.511548 0.803533 0.383794 0.652657
C 0.879067 0.451368 0.207364 0.768483 0.097236
D 0.717062 0.511632 0.164936 0.894755 0.757395
E 0.736824 0.215564 0.546524 0.674393 0.882544

```

**Q9.**

```

import pandas as pd
import numpy as np
column=['a', 'b', 'c', 'd', 'e']
index=['A', 'B', 'C', 'D', 'E']
df1 = pd.DataFrame(np.random.rand(5, 5),
columns = column, index =index)
column=['a', 'b', 'c', 'g', 'h']
print(df1.reindex(column, axis='columns', fill_value = 1.5))
print(df1)
print(df1.reindex(column, axis='columns', fill_value='data missing'))
print(df1)

```

OUTPUT

```

      a      b      c      g      h
A 0.349736 0.520481 0.162386 1.5 1.5
B 0.279018 0.901853 0.911684 1.5 1.5
C 0.763133 0.409622 0.609397 1.5 1.5
D 0.254531 0.177653 0.172417 1.5 1.5
E 0.976517 0.103942 0.963850 1.5 1.5

```

```

      a      b      c      d      e
A 0.349736 0.520481 0.162386 0.541516 0.389136
B 0.279018 0.901853 0.911684 0.507943 0.647398
C 0.763133 0.409622 0.609397 0.098487 0.606802
D 0.254531 0.177653 0.172417 0.192563 0.470302
E 0.976517 0.103942 0.963850 0.333989 0.791105

```

```

      a      b      c      g      h
A 0.349736 0.520481 0.162386 data missing data missing
B 0.279018 0.901853 0.911684 data missing data missing
C 0.763133 0.409622 0.609397 data missing data missing

```

```

D 0.254531 0.177653 0.172417 data missing data missing
E 0.976517 0.103942 0.963850 data missing data missing
    a          b          c          d          e
A 0.349736 0.520481 0.162386 0.541516 0.389136
B 0.279018 0.901853 0.911684 0.507943 0.647398
C 0.763133 0.409622 0.609397 0.098487 0.606802
D 0.254531 0.177653 0.172417 0.192563 0.470302
E 0.976517 0.103942 0.963850 0.333989 0.791105

```

**Q10.**

```

import pandas as pd
info = pd.DataFrame({'A':[1, 5, 3, 4, 2], 'B':[3, 2, 4, 3, 4], 'C':[2, 2, 7, 3, 4], 'D':[4, 3, 6, 12, 7]})
info.reindex(columns=["A", "B", "D", "E"])
info.reindex(columns=["A", "B", "D", "E"], fill_value =37)
print(info)

```

OUTPUT

	A	B	C	D
0	1	3	2	4
1	5	2	2	3
2	3	4	7	6
3	4	3	3	12
4	2	4	4	7

**Q11.**

```

import pandas as pd
df = pd.DataFrame([[1,2,3,4,5],[6,7,8,9,10],[11,12,13,14,15]],index=range(3),
columns=list('abcde'))
print(df)
df2 = df.rename({'a': 'X', 'b': 'Y'}, axis=1) # new method
df2 = df.rename({'a': 'X', 'b': 'Y'}, axis='columns')
df2 = df.rename(columns={'a': 'X', 'b': 'Y'}) # old method
print(df2)
df.rename({'a': 'X', 'b': 'Y'},axis=1, inplace=True)
print(df)

```

OUTPUT

```

a b c d e
0 1 2 3 4 5
1 6 7 8 9 10
2 11 12 13 14 15

X Y c d e
0 1 2 3 4 5
1 6 7 8 9 10
2 11 12 13 14 15

X Y c d e
0 1 2 3 4 5
1 6 7 8 9 10
2 11 12 13 14 15

```

**Q12.**

```

import pandas as pd
import numpy as np
N=20

df = pd.DataFrame({'A': pd.date_range(start='2016-01-01',periods=N,freq='D'),'x':
np.linspace(0,stop=N-1,num=N),'y': np.random.rand(N),'C':
np.random.choice(['Low','Medium','High'],N).tolist(),'D': np.random.normal(100,
10,size=(N)).tolist()})

df_reindexed = df.reindex(index=[0,2,5],columns=['A', 'C', 'B'])

print(df_reindexed)

```

OUTPUT

```

      A      C  B
0 2016-01-01  High NaN
2 2016-01-03   Low NaN
5 2016-01-06  Medium NaN

```

**Q13.**

```

import pandas as pd
import numpy as np

df1 = pd.DataFrame(np.random.randn(6,3),columns=['col1','col2','col3'])

print(df1)

```

```
print ("After renaming the rows andcolumns:")
print(df1.rename(columns={'col1' : 'c1', 'col2' : 'c2'},index = {0 : 'apple', 1 : 'banana',2 : 'durian'}))
```

OUTPUT

	col1	col2	col3
0	3.562122	-0.266875	-0.928852
1	2.155304	1.037614	1.247295
2	-1.002544	-0.085401	-1.458777
3	-1.185789	0.569706	-0.779963
4	0.630365	0.116178	0.252291
5	-0.256166	-0.018474	-0.843920

After renaming the rows andcolumns:

	c1	c2	col3
apple	3.562122	-0.266875	-0.928852
banana	2.155304	1.037614	1.247295
durian	-1.002544	-0.085401	-1.458777
3	-1.185789	0.569706	-0.779963
4	0.630365	0.116178	0.252291
5	-0.256166	-0.018474	-0.843920

**Q14.**

```
import pandas as pd
A = pd.DataFrame({'RollNo':[1,2,3,4,5], 'Name': ['Alex', 'Amy', 'Allen', 'Alice', 'Ayoung'], 'Subject':['S1', 'S2', 'S4', 'S6', 'S5']})
B = pd.DataFrame({'RollNo':[1,2,3,4,5], 'Name': ['Billy', 'Brian', 'Bran', 'Bryce', 'Betty'], 'Subject':['S2', 'S4', 'S3', 'S6', 'S1']})
print(A)
print(B)
```

OUTPUT

	RollNo	Name	Subject
0	1	Alex	S1
1	2	Amy	S2
2	3	Allen	S4
3	4	Alice	S6
4	5	Ayoung	S5

	RollNo	Name	Subject
0	1	Billy	S2
1	2	Brian	S4
2	3	Bran	S3
3	4	Bryce	S6
4	5	Betty	S1

**Q15.**

```
import pandas as pd
```

```
A = pd.DataFrame({'Name': ['Amit', 'Ana', 'Alis', 'Aman',  
'Aadi'], 'subject_id': ['S1', 'S2', 'S4', 'S6', 'S5'], 'Marks_scored': [98, 90, 87, 69, 78]}, index=[1, 2, 3, 4, 5])
```

```
B = pd.DataFrame({'Name': ['Brit', 'Brey', 'Bel', 'Ben',  
'Bali'], 'subject_id': ['S2', 'S4', 'S3', 'S6', 'S5'], 'Marks_scored': [89, 80, 79, 97, 88]}, index=[1, 2, 3, 4, 5])
```

```
print(pd.concat([A, B]))
```

OUTPUT

	Name	subject_id	Marks_scored
1	Amit	S1	98
2	Ana	S2	90
3	Alis	S4	87
4	Aman	S6	69
5	Aadi	S5	78
1	Brit	S2	89
2	Brey	S4	80
3	Bel	S3	79
4	Ben	S6	97
5	Bali	S5	88

**Q16.**

```
import pandas as pd
```

```
import datetime
```

```
d = pd.to_datetime('7/8/2022', dayfirst=True)
```

```
print(d)
```

```
print(pd.period_range('2022-03', periods=8, freq='M'))
```

```
print(pd.timedelta_range(0, periods=9, freq="2H30T"))
```

**OUTPUT**

2022-08-07 00:00:00

```
PeriodIndex(['2022-03', '2022-04', '2022-05', '2022-06', '2022-07', '2022-08',  
            '2022-09', '2022-10'],
```

```
dtype='period[M]')
```

```
TimedeltaIndex(['0 days 00:00:00', '0 days 02:30:00', '0 days 05:00:00',
```

```
'0 days 07:30:00', '0 days 10:00:00', '0 days 12:30:00',
```

```
'0 days 15:00:00', '0 days 17:30:00', '0 days 20:00:00'],
```

```
dtype='timedelta64[ns]', freq='150T')
```



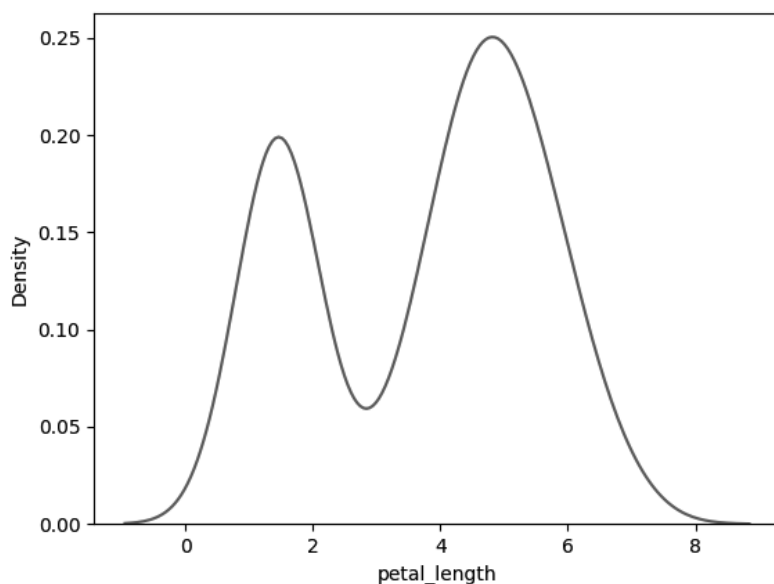
## CHAPTER 6: Plotting Graphs

### Give Output

Q1.

```
import pandas as pd
import seaborn as sb from matplotlib
import pyplot as plt
df = sb.load_dataset('iris') sb.distplot(df['petal_ length'],hist=False)
plt.show()
```

Output

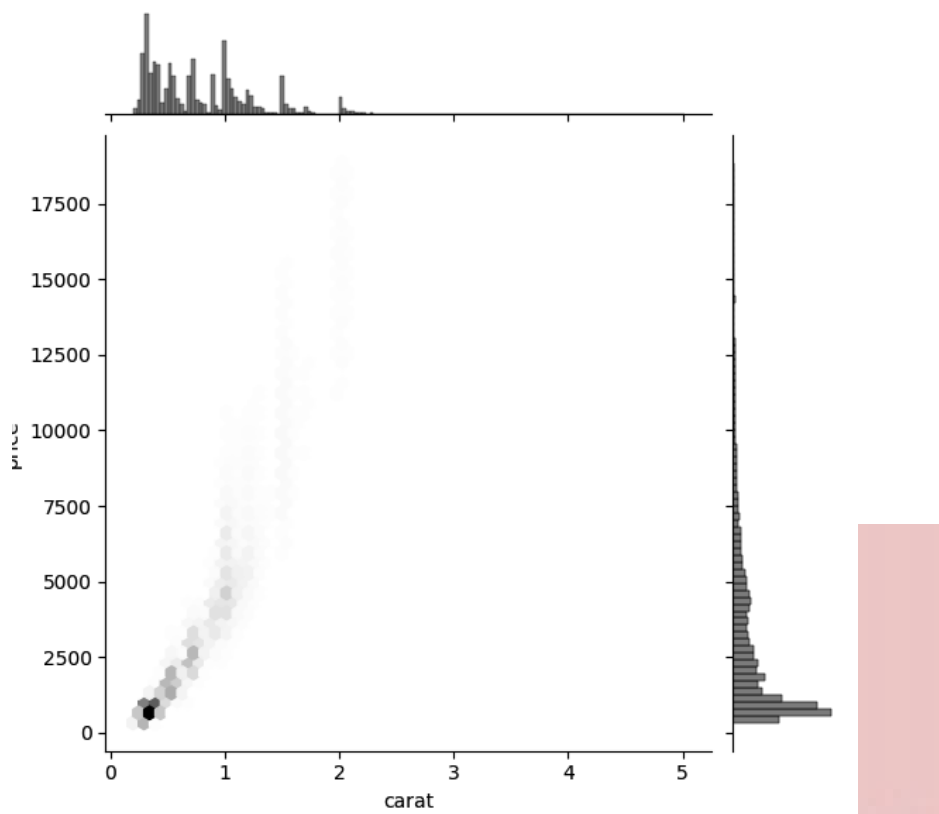


Q2.

```
import pandas as pd
import seaborn as sb from matplotlib
import pyplot as plt
df = sb.load_dataset('diamonds')
sb.jointplot(x = 'carat',y = 'price',data = df,kind = 'hex')
plt.show()
```



output



Graw  
Hill

## CHAPTER 7: File Handling

**Q1.**

```
import os
Files = ['BTech.txt', 'BCA.csv', 'BSc.docx']
for file in Files: print(os.path.join('C:\\Users\\Students', file))
```

OUTPUT

```
C:\Users\Students\BTech.txt
C:\Users\Students\BCA.csv
C:\Users\Students\BSc.docx
```

**Q2.**

```
with open("File.txt", "w") as file: file.write("Greetings to All!!! \n Welcome to the world of
programming\n")
with open("File.txt") as file: print(file.read())
```

OUTPUT

```
Greetings to All!!!
Welcome to the world of programming
```

**Q3.**

```
file=open("File.txt","r")
file.read()
text = file.read()
print(len(text))
file.close()
```

OUTPUT

```
0
```

**Q4.**

```
str="Welcome to Python Programming"
file = open("File.txt","w")
n = file.write(str)
print(n)
file.close()
```

Copyright © 2023 by McGraw Hill Education (India) Private Limited

OUTPUT

29

**Q5.**

```
import csv

fields = ['Name', 'Branch', 'Year', 'CGPA']

rows = [ ['Nikhil', 'COE', '2', '9.0'], ['Sanchit', 'COE', '2', '9.1'], ['Aditya', 'IT', '2', '9.3'], ['Sagar', 'SE', '1', '9.5'], ['Prateek', 'MCE', '3', '7.8'], ['Sahil', 'EP', '2', '9.1']]

filename = "students.csv"

# writing to csv file

with open(filename, 'w') as csvfile:

    csvwriter = csv.writer(csvfile)

    print(csvwriter.writerow(fields))

    print(csvwriter.writerows(rows))
```

OUTPUT

23

None

**Q6.**

```
import csv

# my data rows as dictionary objects

mydict = [{'branch': 'COE', 'cgpa': '9.0', 'name': 'Nikhil', 'year': '2'}, {'branch': 'COE', 'cgpa': '9.1', 'name': 'Sanchit', 'year': '2'}, {'branch': 'IT', 'cgpa': '9.3', 'name': 'Aditya', 'year': '2'}, {'branch': 'SE', 'cgpa': '9.5', 'name': 'Sagar', 'year': '1'}, {'branch': 'MCE', 'cgpa': '7.8', 'name': 'Prateek', 'year': '3'}, {'branch': 'EP', 'cgpa': '9.1', 'name': 'Sahil', 'year': '2'}]

# field names

fields = ['name', 'branch', 'year', 'cgpa']

# name of csv file

filename = "university_records.csv"

# writing to csv file

with open(filename, 'w') as csvfile:

    writer = csv.DictWriter(csvfile, fieldnames = fields)

    print(writer.writeheader())

    print(writer.writerows(mydict))
```

OUTPUT

23

None

**Q7.**

```
import pickle
example_dict = {1:"6",2:"2",3:"f"}
pickle_out = open("dict.pickle","wb")
print(pickle.dump(example_dict, pickle_out))
pickle_out.close()
```

OUTPUT

None

**Q8.**

```
import pickle
pickle_in = open("dict.pickle","rb")
example_dict = pickle.load(pickle_in)
print(example_dict)
print(example_dict[3])
```

OUTPUT

{1: '6', 2: '2', 3: 'f'}

F

**Q9.**

```
import json
student = {'name': 'Rajesh', 'roll': 52, 'subject': ['English', 'Science']}
student_dumped = json.dumps(student)
print(student_dumped)
print(type(student_dumped))
```

OUTPUT

{"name": "Rajesh", "roll": 52, "subject": ["English", "Science"]}

<class 'str'>



## CHAPTER 9: Introduction to Machine Learning

**Q1.**

```
import numpy as np
from sklearn import preprocessing
input_labels = ['red','black','red','green','black','yellow','white']
encoder = preprocessing.LabelEncoder()
encoder.fit(input_labels)
test_labels = ['green','red', 'black']
encoded_values = encoder.transform(test_labels)
print("\nLabels =", test_labels)
print("Encoded values =",list(encoded_values))
encoded_values = [3,0,4,1]
decoded_list = encoder.inverse_transform(encoded_values)
print("\nEncoded values =",encoded_values)
print("\nDecoded labels =",list(decoded_list))
```

OUTPUT

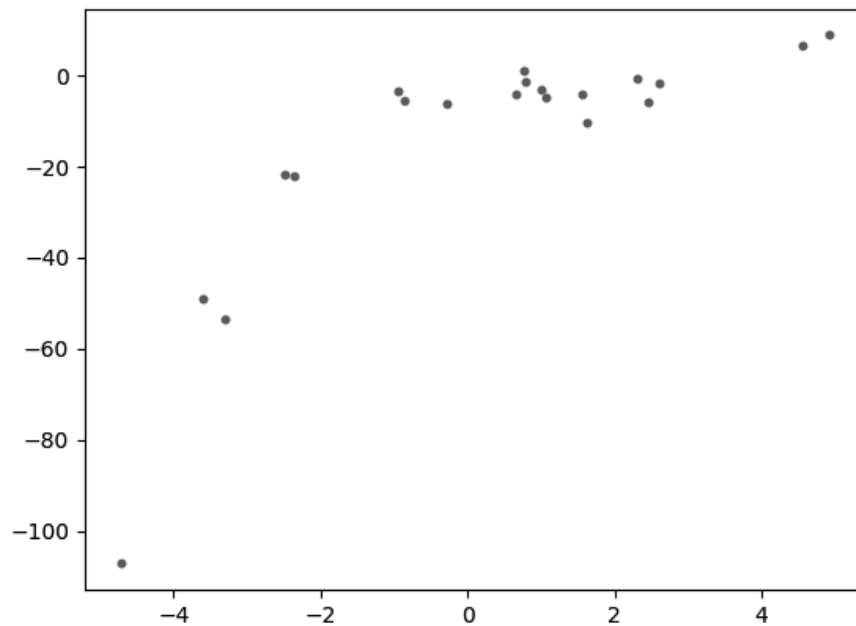
```
Labels = ['green', 'red', 'black']
Encoded values = [1, 2, 0]
Encoded values = [3, 0, 4, 1]
Decoded labels = ['white', 'black', 'yellow', 'green']
```

## CHAPTER 10: Regression Analysis in Machine Learning

Q1.

```
import numpy as np
import matplotlib.pyplot as plt
np.random.seed(0)
x = 2 - 3 * np.random.normal(0, 1, 20)
y = x - 2 * (x ** 2) + 0.5 * (x ** 3) + np.random.normal(-3, 3, 20)
plt.scatter(x, y, s=10)
plt.show()
```

OUTPUT



## CHAPTER 12: Advanced Learning

### GIVE OUTPUT

**Q1.**

```
import nltk

word_data = " Natural Language Processing (NLP) is a process of manipulating or understanding
the text or speech by any software or machine. An analogy is that humans interact and
understand each other's views and respond with the appropriate answer. In NLP, this
interaction, understanding, and response are made by a computer instead of a human."

nltk_tokens = nltk.word_tokenize(word_data)

print (nltk_tokens)
```

#### OUTPUT

```
['Natural', 'Language', 'Processing', '(', 'NLP', ')', 'is', 'a', 'process', 'of', 'manipulating', 'or',
'understanding', 'the', 'text', 'or', 'speech', 'by', 'any', 'software', 'or', 'machine', '.', 'An', 'analogy',
'is', 'that', 'humans', 'interact', 'and', 'understand', 'each', 'other', "'s", 'views', 'and', 'respond',
'with', 'the', 'appropriate', 'answer', '.', 'In', 'NLP', ',', 'this', 'interaction', ',', 'understanding', ',',
'and', 'response', 'are', 'made', 'by', 'a', 'computer', 'instead', 'of', 'a', 'human', '.']
```

**Q2.**

```
import nltk

sentence_data = "Natural Language Processing (NLP) is a process of manipulating or
understanding the text or speech by any software or machine. An analogy is that humans
interact and understand each other's views and respond with the appropriate answer. In NLP,
this interaction, understanding, and response are made by a computer instead of a human."

nltk_tokens = nltk.sent_tokenize(sentence_data)

print (nltk_tokens)
```

#### OUTPUT

```
['Natural Language Processing (NLP) is a process of manipulating or understanding the text or
speech by any software or machine.', 'An analogy is that humans interact and understand each
other's views and respond with the appropriate answer.', 'In NLP, this interaction,
understanding, and response are made by a computer instead of a human.']
```

**Q3.**

```
import nltk

from nltk.stem.porter import PorterStemmer

porter_stemmer = PorterStemmer()

word_data = "Natural Language Processing (NLP) is a process of manipulating or understanding
the text or speech by any software or machine. An analogy is that humans interact and
```

Copyright © 2023 by McGraw Hill Education (India) Private Limited

understand each other's views and respond with the appropriate answer. In NLP, this interaction, understanding, and response are made by a computer instead of a human."

# First Word tokenization

```
nlk_tokens = nltk.word_tokenize(word_data)
```

#Next find the roots of the word

```
for w in nlk_tokens:print("Actual: %s Stem: %s" %(w,porter_stemmer.stem(w)))
```

OUTPUT

Actual: Natural Stem: natur

Actual: Language Stem: languag

Actual: Processing Stem: process

Actual: ( Stem: (

Actual: NLP Stem: nlp

Actual: ) Stem: )

Actual: is Stem: is

Actual: a Stem: a

Actual: process Stem: process

Actual: of Stem: of

Actual: manipulating Stem: manipul

Actual: or Stem: or

Actual: understanding Stem: understand

Actual: the Stem: the

Actual: text Stem: text

Actual: or Stem: or

Actual: speech Stem: speech

Actual: by Stem: by

Actual: any Stem: ani

Actual: software Stem: softwar

Actual: or Stem: or

Actual: machine Stem: machin

Actual: . Stem: .

Actual: An Stem: an

Actual: analogy Stem: analog

Actual: is Stem: is

Actual: that Stem: that



Actual: humans Stem: human

Actual: interact Stem: interact

Actual: and Stem: and

Actual: understand Stem: understand

Actual: each Stem: each

Actual: other Stem: other

Actual: ' Stem: '

Actual: s Stem: s

Actual: views Stem: view

Actual: and Stem: and

Actual: respond Stem: respond

Actual: with Stem: with

Actual: the Stem: the

Actual: appropriate Stem: appropri

Actual: answer Stem: answer

Actual: . Stem: .

Actual: In Stem: in

Actual: NLP Stem: nlp

Actual: , Stem: ,

Actual: this Stem: thi

Actual: interaction Stem: interact

Actual: , Stem: ,

Actual: understanding Stem: understand

Actual: , Stem: ,

Actual: and Stem: and

Actual: response Stem: respons

Actual: are Stem: are

Actual: made Stem: made

Actual: by Stem: by

Actual: a Stem: a

Actual: computer Stem: comput

Actual: instead Stem: instead

Actual: of Stem: of

Actual: a Stem: a

Actual: human Stem: human

Actual: . Stem: .

**Q4.**

```
import nltk
from nltk.stem import WordNetLemmatizer
wordnet_lemmatizer = WordNetLemmatizer()
word_data = "Natural Language Processing (NLP) is a process of manipulating or understanding
the text or speech by any software or machine. An analogy is that humans interact and
understand each other's views and respond with the appropriate answer. In NLP, this
interaction, understanding, and response are made by a computer instead of a human."
nltk_tokens = nltk.word_tokenize(word_data)
for w in nltk_tokens: print("Actual: %s Lemma: %s" % (w, wordnet_lemmatizer.lemmatize(w)))
```

OUTPUT

Actual: Natural Lemma: Natural

Actual: Language Lemma: Language

Actual: Processing Lemma: Processing

Actual: ( Lemma: (

Actual: NLP Lemma: NLP

Actual: ) Lemma: )

Actual: is Lemma: is

Actual: a Lemma: a

Actual: process Lemma: process

Actual: of Lemma: of

Actual: manipulating Lemma: manipulating

Actual: or Lemma: or

Actual: understanding Lemma: understanding

Actual: the Lemma: the

Actual: text Lemma: text

Actual: or Lemma: or

Actual: speech Lemma: speech

Actual: by Lemma: by

Actual: any Lemma: any

Actual: software Lemma: software

Actual: or Lemma: or

Copyright © 2023 by McGraw Hill Education (India) Private Limited

Actual: machine Lemma:machine

Actual: . Lemma:.

Actual: An Lemma:An

Actual: analogy Lemma:analogy

Actual: is Lemma:is

Actual: that Lemma:that

Actual: humans Lemma:human

Actual: interact Lemma:interact

Actual: and Lemma:and

Actual: understand Lemma:understand

Actual: each Lemma:each

Actual: other Lemma:other

Actual: ' Lemma:'

Actual: s Lemma:s

Actual: views Lemma:view

Actual: and Lemma:and

Actual: respond Lemma:respond

Actual: with Lemma:with

Actual: the Lemma:the

Actual: appropriate Lemma:appropriate

Actual: answer Lemma:answer

Actual: . Lemma:.

Actual: In Lemma:In

Actual: NLP Lemma:NLP

Actual: , Lemma:,

Actual: this Lemma:this

Actual: interaction Lemma:interaction

Actual: , Lemma:,

Actual: understanding Lemma:understanding

Actual: , Lemma:,

Actual: and Lemma:and

Actual: response Lemma:response

Actual: are Lemma:are

Actual: made Lemma:made

Actual: by Lemma:by

Actual: a Lemma:a

Actual: computer Lemma:computer

Actual: instead Lemma:instead

Actual: of Lemma:of

Actual: a Lemma:a

Actual: human Lemma:human

Actual: . Lemma:.

