Program 1: Predict the output of the following Python Programs.

```
class Acc:
  def __init__(self, id):
    self.id = id
    id = 555
acc = Acc(111)
print(acc.id)
```

Explanation: Instantiation of the class "Acc" automatically calls the method __init__ and passes the object as the self parameter. 111 is assigned to data attribute of the object called id.

The value "555" is not retained in the object as it is not assigned to a data attribute of the class/object. So, the output of the program is "111"

Program 2: Predict the output of the following Python Programs.

```
for i in range(2):
  print (i)
for i in range(4,6):
  print(i)
```

Explanation: If only single argument is passed to the range method, Python considers this argument as the end of the range and the default start value of range is 0. So, it will print all the numbers starting from 0 and before the supplied argument.

For the second for loop the starting value is explicitly supplied as 4 and ending is 5.

Program 3: Predict the output of following Python Programs

```
values = [1, 2, 3, 4]
numbers = set(values)
def checknums(num):
  if num in numbers:
    return True
  else:
    return False
for i infilter(checknums, values):
  print(i)
```

Explanation: The function "filter" will return all items from list values that return True when passed to the function "check it". "check it" will check if the value is in the set. Since all the numbers in the set come from the values list, all of the original values in the list will return True.

Program 4: Predict the output of the following Python Programs

```
counter = {}
def addToCounter(country):
   if country in counter:
counter[country] += 1
   else:
counter[country] = 1
addToCounter('China')
addToCounter('Japan')
addToCounter('china')
print len(counter)
```

Explanation: The task of "len" function is to return number of keys in a dictionary. Here 3 keys are added to the dictionary "country" using the "add to counter" function.

Please note carefully – The keys to a dictionary are case sensitive.

Program 5: Predict the output of following Python Programs

```
def dtFunction():
   "DataTrained is cool website for boosting up technical skills"
   return 1
print (dtFunction.__doc__[15:19] )
```

Explanation:

There is a docstring defined for this method, by putting a string on the first line after the start of the function definition. The docstring can be referenced using the __doc__ attribute of the function.

And hence it prints the indexed string.

Program 6: Predict the output of the following Python Programs

```
class A(object):
   val = 1
class B(A):
   pass
class C(A):
   pass
print(A.val, B.val, C.val)
B.val = 2
print(A.val, B.val, C.val)
A.val = 3
print(A.val, B.val, C.val)
```

E**xplanation:**

In Python, class variables are internally handled as dictionaries. If a variable name is not found in the dictionary of the current class, the class hierarchy (i.e., its parent classes) are searched until the referenced variable name is found, if the variable is not found error is being thrown.

So, in the above program the first call to print() prints the initialized value i.e, 1.

In the second call since B. val is set to 2, the output is 1 2 1.

The last output 3 2 3 may be surprising. Instead of 3 3 3, here B.val reflects 2 instead of 3 since it is overridden earlier.

Program 7: Predict the output of the following Python Programs

```
check1 = ['Learn', 'Quiz', 'Practice', 'Contribute']
check2 = check1
check3 = check1[:]
check2[0] = 'Code'
check3[1] = 'Mcq'
count = 0
for c in (check1, check2, check3):
   if c[0] == 'Code':
      count += 1
   if c[1] == 'Mcq':
      count += 10
```

```
print(count)
```

Explanation:

When assigning check1 to check2, we create a second reference to the same list. Changes to check2 affects check1. When assigning the slice of all elements in check1 to check3, we are creating a full copy of check1 which can be modified independently (i.e, any change in check3 will not affect check1).

So, while checking check1 'Code' gets matched and count increases to 1, but Mcq doest gets matched since its available only in check3.

Now checking check2 here also 'Code' gets matched resulting in count value to 2.

Finally while checking check3 which is separate from both check1 and check2 here only Mcq gets matched and count becomes 12.

Program 8: Predict the output of the following Python Programs

```
def dataTr(x,l=[]):
  for i in range(x):
    l.append(i*i)
  print(l)
dataTr(2)
dataTr(3,[3,2,1])
dataTr(3)
```

Explanation:

The first function call should be fairly obvious, the loop appends 0 and then 1 to the empty list, l. l is a name for a variable that points to a list stored in memory. The second call starts off by creating a new list in a new block of memory. l then refers to this new list. It then appends 0, 1 and 4 to this new list. So that's great. The third function call is the weird one. It uses the original list stored in the original memory block. That is why it starts off with 0 and 1.

Program 9: Which of the options below could possibly be the output of the following program?

```
from random import randrange
L = list()
for x in range(5):
L.append(randrange(0, 100, 2)-10)
# Choose which of outputs below are valid for this code.
```

```
print(L)
```

a) [-8, 88, 8, 58, 0]

b) [-8, 81, 18, 46, 0]

c) [-7, 88, 8, 58, 0]

d) [-8, 88, 94, 58, 0]

Ans. (a)

Explanation: The for loop will result in appending 5 elements to list L. Range of the elements lies from [0, 98] – 10 = [-10, 88], which rules out option (d). The upper range is 98 because the step size is 2, thus option (c) and (b) are invalid. Also, note that each time you may not get the same output or the one in the options as the function is random.

Program 10: What is the output of the following program?

```
from math import *

a = 2.13

b = 3.7777

c = -3.12

print(int(a), floor(b), ceil(c), fabs(c))
```

a) 2 3 -4 3

b) 2 3 -3 3.12

c) 2 4 -3 3

d) 2 3 -4 3.12

Ans. (b)

Explanation: int() returns the integer value of a number, int(2.13) = 2. floor() returns the largest integer lesser or equal to the number, floor(3.777) = 3. ceil() returns smallest integer greater or equal to the number, ceil(-3.12) = -3. fabs() return the modulus of the number, thus fabs(-3.12) = 3.12.

P**rogram 11: What is the output of the following program?**

```
import re

p = re.compile('\d+')

print(p.findall("I met him once at 11 A.M. on 4th July 1886"), end = " ")

p = re.compile('\d')

print(p.findall("I went to him at 11 A.M."))
```

a) [’11’, ‘4’, ‘1886’, ’11’]

b) [‘1141886’] [‘1’, ‘1’]

c) [’11’, ‘4’, ‘1886’] [’11’]

d) [’11’, ‘4’, ‘1886’] [‘1’, ‘1’]

Ans. (d)

Explanation: \d is equivalent to [0-9] and \d+ will match a group on [0-9], group of one or greater size. In first statement, group of digits are 11, 4, 1886. In the second statement, \d will treat each each digit as different entity, thus 1, 1.

Program 12: What is the output of the following program?

```
import re

print(re.sub('a', '**', 'DataTrainers', flags = re.IGNORECASE), end = " ")

print(re.sub('tr', '**', 'DataTrainers'))
```

a) D**t**Tr**iners Data**ainers

b) D*t*Tr*iners Data**ainers

c) D*t*Tr*iners Data*ainers

d) TypeError: ‘str’ object does not support item assignment

Ans. (a)

Explanation: In the first print statement, all ‘a’ will be replaced ‘’, and case is igonred. Case is not igonred in 2nd statement, thus ‘ge’ will be replaced but not ‘Ge’.**

Program 13: Which of the options below could possibly be the output of the following program?

```
import math

import random

L = [1, 2, 30000000000000]

for x in range(3):

  L[x] = math.sqrt(L[x])

# random.choices() is available on Python 3.6.1 only.

string = random.choices(["apple", "carrot", "pineapple"], L, k = 1)

print(string)
```

a) [‘pineapple’]

b) [‘apple’]

c) ‘pineapple’

d) both a and b

Ans. **(d)**

Explanation: Two modules math and random are used, L after the for loop will be [1.0, 1.4142135623730951, 5477225.575051662]. choices() has a choice as 1st parameter and their weights as the second parameter, k is the number valued needed from choice. The answer will come out to ‘pineapple’ almost always due to its the weight but ‘apple’ and ‘carrot’ may turn out to be the output at times.

Program 14: What is the output of the following program?

```
D = {1 : 1, 2 : '2', '1' : 1, '2' : 3}

D['1'] = 2

print(D[D[D[str(D[1])]]])
```

a) 2

b) 3

c) ‘2’

d) KeyError

Ans. (b)

Explanation: Simple key-value pair is used recursively, D[1] = 1, str(1) = ‘1’. So, D[str(D[1])] = D[‘1’] = 2, D[2] = ‘2’ and D[‘2’] = 3

Program 15: What is the output of the following program?

```
D = dict()

for x in enumerate(range(2)):

  D[x[0]] = x[1]

  D[x[1]+7] = x[0]

print(D)
```

a) KeyError

b) {0: 1, 7: 0, 1: 1, 8: 0}

c) {0: 0, 7: 0, 1: 1, 8: 1}

d) {1: 1, 7: 2, 0: 1, 8: 1}

Ans. **(c)**

Explanation: enumerate() will return a tuple, the loop will have x = (0, 0), (1, 1). Thus D[0] = 0, D[1] = 1, D[0 + 7] = D[7] = 0 and D[1 + 7] = D[8] = 1.

Note: Dictionary is unordered, so the sequence of the key-value pair may differ in each output.

Program 16: Which of the options below could possibly be the output of the following program?

```
D = {1 : [1, 2, 3], 2: (4, 6, 8)}

D[1].append(4)

print(D[1], end = " ")

L = list(D[2])

L.append(10)

D[2] = tuple(L)

print(D[2])
```

a) [1, 2, 3, 4] [4, 6, 8, 10]

b) [1, 2, 3] (4, 6, 8)

c) '[1, 2, 3, 4] TypeError: tuples are immutable

d) [1, 2, 3, 4] (4, 6, 8, 10)

Ans. (d)

Explanation: In the first part key-value indexing is used and 4 is appended into the list. As tuples are immutable, in the second part the tuple is converted into a list, valued 10 is added and then converted back to list.

Program 17: What is the output of the following program?

```
D = dict()

for i in range (3):

   for j in range(2):

      D[i] = j

print(D)
```

a) {0: 0, 1: 0, 2: 0}

b) {0: 1, 1: 1, 2: 1}

c) {0: 0, 1: 0, 2: 0, 0: 1, 1: 1, 2: 1}

d) TypeError: Immutable object

Ans. (b)

Explanation: 1st loop will give 3 values to i 0, 1 and 2. In the empty dictionary, valued are added and overwrited in j loop, for eg. D[0] = [0] becomes D[0] = 1, due to overwriting.

Program 18: What is the output of the following program?

data = [2, 3, 9]

temp = [[x for x in[data]] for x in range(3)]

print (temp)

a) [[[2, 3, 9]], [[2, 3, 9]], [[2, 3, 9]]]

b) [[2, 3, 9], [2, 3, 9], [2, 3, 9]]

c) [[[2, 3, 9]], [[2, 3, 9]]]

d) None of these

Ans. (a)

Explanation: [x for x in[data] returns a new list copying the values in the list data and the outer for statement prints the newly created list 3 times.

Program 19: What is the output of the following program?

data = [x for x in range(5)]

temp = [x for x in range(7) if x in data and x%2==0]

print(temp)

a) [0, 2, 4, 6]

b) [0, 2, 4]

c) [0, 1, 2, 3, 4, 5]

d) Runtime error

Ans. (b)

Explanation: The is statement checks whether the value lies in list data and if it does whether it's divisible by 2. It does so for x in (0, 7).

Program 19: What is the output of the following program?

temp = ['Data', 'for', 'Peoples']

arr = [i[0].upper() for i in temp]

print(arr)

a) [‘D’, ‘F’, ‘P’]

b) [‘DATA’]

c) [‘DATA’, ‘FOR’, ‘PEOPLES’]

d) Compilation error

A**ns. (a)**

Explanation: The variable i is used to iterating over each element in list temp. i[0] represent the character at 0th index of i and .upper() function is used to capitalize the character present at i[0].

Program 20: Whats is the output of the following program?

```
temp = 'Treat 22536 for 445 Geeks'

data = [x for x in (int(x) for x in temp if x.isdigit()) if x%2 == 0]

print(data)
```

a) [2, 2, 6, 4, 4]

b) Compilation error

c) Runtime error

d) [‘2’, ‘2’, ‘5’, ‘3’, ‘6’, ‘4’, ‘4’, ‘5’]

Ans. (a)

Explanation: This is an example of nested list comprehension. The inner list created contains a list of integers in temp. The outer list only procures that x which are a multiple of 2.

Program 21: What is the output of the following program?

```
data = [x for x in (x for x in 'Geeks 22966 for Geeks' if x.isdigit()) if

(x in ([x for x in range(20)]))]

print(data)
```

a) [2, 2, 9, 6, 6]

b) []

c) Compilation error

d) Runtime error

Ans. (b)

Explanation: Since here x have not been converted to int, the condition in the if statement fails and therefore, the list remains empty.

What is the output of the following?

Program 22:

```
i = 1
while True:
   if i % 0O7 == 0:
      break
   print(i)
   i += 1
```

1. 1 2 3 4 5 6.
2. 1 2 3 4 5 6 7.
3. error.
4. None of these

Ans. (a)

Explanation: The loop will terminate when i will be equal to 7.

Program 22: What is the output of the following program?

```
str1 = '{2}, {1} and {0}'.format('a', 'b', 'c')
str2 = '{0}{1}{0}'.format('abra', 'cad')
print(str1, str2)
```

a) c, b and a abracad0

b) a, b and c abracadabra

c) a, b and c abracadcad

d) c, b and a abracadabra

Ans. (d)

Explanation: String function format takes a format string and an arbitrary set of positional and keyword arguments. For str1 'a' has index 2, 'b' index 1 and 'c' index 0. str2 has only two indices 0 and 1. Index 0 is used twice at 1st and 3rd time.

Program 23: What is the output of the following program?

```
a = 2
b = '3.77'
c = -8
```

```
str1 = '{0:.4f} {0:3d} {2} {1}'.format(a, b, c)

print(str1)
```

a) 2.0000 2 -8 3.77

b) 2 3.77 -8 3.77

c) 2.000 3 -8 3.77

d) 2.000 2 8 3.77

Ans. (a)

Explanation: At Index 0, integer a is formatted into a float with 4 decimal points, thus 2.0000. At Index 0, a = 2 is formatted into a integer, thus it remains to 2. Index 2 and 1 values are picked next, which are -8 and 3.77 respectively.

Program 24: What is the output of the following program?

```
import string

import string

Line1 = "And Then There Were None"

Line2 = "Famous In Love"

Line3 = "Famous Were The Kol And Klaus"

Line4 = Line1 + Line2 + Line3

print(string.find(Line1, 'Were'), string.count((Line4), 'And'))
```

a) True 1

b) 15 2

c) (15, 2)

d) True 2

Ans. (c)

Explanation: 'Were' is at Index 15 in Line1, find() returns the index of substring if found in the string Line1. count() returns the total number of occurrences of the substring. Line4 is concatenated string from Line1, Line2 and Line3. This code works well with Python v2.x, as some string functions are deprecated in Python v3.x.

Program 25: What is the output of the following program?

```
line = "I'll come by then."

eline = ""

for i in line:
    eline += chr(ord(i)+3)
print(eline)
```

a) L*oo frph e| wkhq1

b) L*oo#frph#e|#wkhq1

c) l*oo@frph@e|$wkhq1

d) O*oo#Frph#E|#wKhq1

Ans. (b)

Explanation: This piece of code ciphers the plain text. Each character is moved to its 3rd next character by increasing the ASCII value. 'I' becomes 'L', thus option (c) and (d) are ruled out. ' ' has ASCII value of 32, thus it'll become 35('#'), thus option (a) is ruled out as, ' ' cannot remain to be ' ' in the ciphered text.