PYTHON REVISION TOUR

VARIABLES

A variable is a container that stores values that you can access or change. The purpose of using variables is to allow the stored values to be used later on. Any object or variable in python that refers to a value at a particular memory location and possesses three components.

- A value – it represents any number or a letter or a string. To assign any value to a variable ,use assignment operator(=)
- An identity- it refers to the address of the variable in memory which does not change once created. To retrieve the address (identity) of a variable , the command used is

  ```
  >>> id(varname)
  ```
- A type – we are not required to explicitly declare a variable with its type. Whenever we declare a variable with some value ,python automatically allocates the relevant data type associated with it.

DATA TYPES

Numbers	Integer	Boolean
	Floating point	
	Complex	
None		
Sequences	Strings Tuple List	
Sets		
Mappings	Dictionary	

Integer – to store whole numbers

Float – to store number with fraction. They can be represented in scientific notation where the uppercase or lowercase letter ‘e’ signifies the 10th power

>>>3.8e2 ---------------→380.0

>>>7.2e1 ---------------→72.0

Complex numbers – are pairs of real and imaginary numbers. They take the form a+bj

Boolean – is used in situations where comparison to be made always result in either a true or a false value.

```
>>>print(5==6*2)  --------------------→False
>>>print(6<3*5-4) -----------------→ True
```

None – is a special data type with an unidentified value

Sequence – is an ordered collection of items , indexed by integers (both positive as well as negative). The three types of sequence data types available in python are strings, lists and tuples.

Sets- is an unordered collection of values of any type with no duplicate entry. It is immutable

Mappings- is unordered and mutable, it represent data in key-value pairs and accessed using keys, which are immutable and are enclosed in {}

KEYWORDS

These are reserved words used by a python interpreter to recognize the structure of a program. As these words have specific meanings for the interpreter they cannot be used as a variable name or for any other purpose. For checking /displaying the list of keywords available in python

```
 import keyword
>>>print(keyword.kwlist)
```

OPERATORS

- Arithmetic operators (+(unary),-(unary),+(binary),-(binary),*,/,//,%,**)
- Assignment operator (=,+=,-=,*=,/=,**=,%=,//=)
- Relational operator (==,!=,<,>,<=,>=)
- Logical operators (or,and,not)

INPUT AND OUTPUT

1. input() – accepts and returns the user's input as a string and stores it in the variable which is assigned with the assignment operator.
2. eval() – method takes a string as an argument ,evaluates this string as a number, and returns the numeric result. If the given argument is not a string or if it cannot be evaluated as a number then eval() results in an error.

3. print()- which is used to display the specified content on the screen.

COMMENTS

A comment in python starts with a # symbol anywhere in a line and extends till the end of the line. In case of multiline comments , we can use triple quoted strings (single or double) ‘’’…..’’’ or “””….”””

FLOW OF EXECUTION

1. sequential statements
2. Selection/ Conditional statements(if,if..elif)
3. Iteration/ looping constructs (for,while)

STRINGS

These are immutable.

Operator	Name	Description
+	Concatenation	Adds or joins two strings
*	Repetition	Concatenates multiple copies of the same string
in/not in	Membership	Returns true if a character exists/not exists in the given string
[:]	Range(start,stop[,step])	Extracts the characters from the given range
[]	Slice[n:m]	Extracts the characters from the given index

Built in string methods

Method	Description	Example

isalpha()	Returns True if the string contains only letters, otherwise returns False	St=”good”
		>>>print(St.isalpha())
		True
		St1=”This is a string”
		>>>print(St1.isalpha())
		False

isdigit()	Returns True if the string contains only digits,	St=”1234”
		>>>print(St.isdigit())
		True
		St1=”This is 1 string”

	otherwise returns False	>>>print(St1.isdigit()) False

lower()	Converts all the uppercase letters in the string to lowercase	St=”Python”
		>>>print(st.lower())
		Python

islower()		
upper()		
isupper()		

lstrip() or lstrip(chars)	Returns the string after removing the space(s) from the left of the string	St=”Green Revolution”
		>>>print(St.lstrip(“Gr”))
		een Revolution
		st1=” Green”
		>>>print(st1.lstrip())
		Green

rstrip() or rstrip(chars)		
isspace()		

istitle()	Returns True if string is properly “titlecased” otherwise returns False	St=”All Learn Python”
		>>>print(St.istitle())
		True
		St1=”All learn Python”
		>>>print(St1.istitle())
		False

join(sequence)	Returns a string in which the string elements have been joined by a string separator	St=”12345”
		S=”-“
		>>>S.join(St)
		‘1-2-3-4-5’

swapcase()	This method converts and returns all uppercase characters to lowercase and vice versa.	St=”weLComE”
		>>>St.swapcase()
		WElcOMe

partition(separator)	Is used to split the given string using the specified separator and	S=xyz@gmail.com
		>>>S.partition(‘ ‘)
		(xyz@gmail.com,’ ‘’, ‘’)
		M=”hardworkpays”

	return a tuple with three basic parts: substring before the separator, separator itself, a substring after the separator If the separator is not found it returns the string itself, followed by two empty strings within the parentheses as a tuple	>>>M.partition(‘work’) (‘hard’,’work’,’pays’)
Python provides two functions for character encoding		

ord()	Returns the ASCII code of the character	Ch=’b’
		>>>ord(Ch)
		98

chr()	Returns the character represented by the inputted ASCII number	>>>chr(66) B

LISTS

A list is a data type that can be used to store any type and number of variables and information. The values in the list are called elements or items or list members.

A list in python is formed by enclosing the values inside [ ] . Lists are mutable ie the values in a list can be changed or modified and can be accessed using index value enclosed in square brackets.

Syntax

Listname= [item1,item2,….]

Example

L=[10,20,30,40,50]

0	1	2	3	4	(+ive index)
10	20	30	40	50	
-5	-4	-3	-2	-1	(-ve index)

LIST COMPREHENSION

List comprehension is an elegant and concise way of creating a new list from an existing list in Python. List comprehension consists of an expression followed by 'for statement' inside square brackets.

Syntax

Newlist=[expression for item in list if condition]

L1=[I*I for I in range(5) if I%2==0]

\>>>print(L1)

[0,2,4]

LIST SLICING

List slices are the sub parts of a list extracted out. List slices can be created using indexes. Slicing is used to retrieve a subset of values. A slice of a list is basically its sub list. When we enter a range that we want to extract is called range slicing.

Syntax

List[start:stop:step]

Step is optional default value is 1

L=[100,200,300,400,500,600,700,800,900]

0	1	2	3	4	5	6	7	8
100	200	300	400	500	600	700	800	900
-9	-8	-7	-6	-5	-4	-3	-2	-1

>>>L[5:]

```
[600,700,800,900]
>>>L[2:6]
[300,400,500,600]
>>>L[-9:-5]
[100,200,300,400]
>>>L[::-1]
[900,800,700,600,500,400,300,200,100]
L1=[‘MY LAB’,[1,2,3],’Y’,(3,4,6),’TABLE’,50]
```

0	1	2	3	4	5
MY LAB	1 2 3 0 1 2	Y	(3,4,6) 0 1 2	TABLE	50

```
>>>L1[2:3]
[‘Y’]
>>>L1[1:2]
[[1,2,3]]
>>>L1[3][1]
```

LIST FUNCTIONS

SNO	FUNCTION	DESCRIPTION
1	cmp(list1,list2)	Compares elements from both the lists
2	len(list)	Returns the total length of the list
3	max(list)	Returns the item with maximum value in the list
4	min(list)	Returns the item with minimum value in the list
5	list(seq)	Converts a tuple into list
6	sum(list)	Sums up all the numerical value in the list

LIST METHODS

Method	Example	Description

append(item)	>>>L=[10,20]
	>>>L.append(30)
	>>>print(L)
	10,20,30
		Adds item to the end of the list

extend(items)	>>>L1=[1,2,5,4]
	>>>A = [100, 90, 80, 50]
	>>> L1. extend (A)
	>>> print(L1)
	[1,2,5,4,100,90,80,50]
	>>>print(A)
	[100,90,80,50]
		For adding more than one element, extend ( ) method can be used, this can also be used to add elements of another list to the existing one.

index(item)	L=[10,20,30]
	>>>print(L.index(20))
	1
	>>>print(L.index(50))
	ValueError
		Returns the index of the first element whose value is equal to the first item. A ValueError exception is raised if item is not found in then list.

insert(index,item)	>>>L=[10,20,30,40]
	>>>L.insert(2,25)
	>>>print(L)
	>>>L.insert(5,60)
	>>>print(L)
	>>>L.insert(-9,5)
	>>>print(L)
	[10, 20, 25, 30, 40]
	[10, 20, 25, 30, 40, 60]
	[5, 10, 20, 25, 30, 40, 60]
		Inserts item into the list at the specified index. When an item is inserted into a list, the list is expanded in size to accommodate the new item. The item that was previously at the specified index, and all the item after it, are shifted by one position towards the end of the list. No exceptions will occur if we specify an invalid index. If we specify an index beyond the end of the list, the item will be added at the end. If we use a negative index that specifies an invalid position the item will be

		inserted at the beginning of the list.

sort()	L=[2,4,1,6,7,9,5]
	L.sort() #ascending
	Print(L)
	L.sort(reverse=True) #descending
	print(L)
		Sorts the items in the list so that they appear in ascending order.

reverse()	L.reverse()	Reverse the order of the items

pop()	>>> L1 = [1, 2, 5, 4, 70, 10, 90, 80, 50]
	>>> a= L1.pop (1) # here the element deleted will be returned to **‘a’**
	>>> print L1
	[1, 5, 4, 70, 10, 90, 80, 50]
	>>> print a
	2
	If no index value is provided in pop ( ), then last element is deleted.
	>>>L1.pop ( )
	50
		It removes the element from the specified index, and also return the element which was removed.
		List.pop ([index])

remove()	>>> L1. remove (90) will remove the value 90 from the list	

UPDATING LIST ELEMENTS

Updating an element of list is, accomplished by accessing the element & modifying its value in place. It is possible to modify a single element or a part of list. For first type, we use index to access single element and for second type, list slice is used. We have seen examples of updations of an element of list. Lets update a slice.

Example

```
>>> L1 [1:2] = [10, 20]
>>> print L1
```

will produce

```
[1, 10, 20, 4, 70, 100, 90, 80, 50]
```

Example

```
>>>A=[10, 20, 30, 40]
>>>A [1:4] = [100]
>>>print A
```

will produce

```
[10, 100]
```

As lists are sequences, they support many operations of strings. For example, operator **+ ,**& ,***** results in concatenation & repetition of lists. Use of these operators generate a new list.

Example

```
>>> a= L1+L2
```

will produce a 3rd list **a** containing elements from L1 & then L2. **a** will contain

```
[1, 10, 20, 4, 70, 100, 90, 80, 50, “Delhi”, “Chennai”, “Mumbai”]
```

Example

```
>>> [1, 2, 3] + [4, 5, 6]
[1, 2, 3, 4, 5, 6]
```

Example

```
>>> b = L1*2
>>> print b
[[1, 10, 20, 4, 70, 100, 90, 80, 50, 1, 10, 20, 4, 70, 100, 90, 80, 50]
```

Example

```
>>> [„Hi!‟]* 3
[„Hi!‟, „Hi!‟, „Hi!‟]
```

It is important to know that **‘+’** operator in lists expects the same type of sequence on both the sides otherwise you get a type error. If you want to concatenate a list and string, either you have to convert the list to string or string to list**.**

Example

```
>>> str([11, 12]) + “34” or >>>“[11,12]” + “34”
„[11, 12] 34‟
>>> [11, 12] + list (“34”) or >>>[11, 12] + [“3”, “4”]
[11, 12, „3‟, „4‟]
```

Deleting Elements

It is possible to delete/remove element(s) from the list. There are many ways of doing

so:
(i) If index is known, we can use pop ( ) or del
(ii) If the element is known, not the index, remove ( ) can be used.
(iii) To remove more than one element, del ( ) with list slice can be used.
(iv) Using assignment operator

del removes the specified element from the list, but does not return the deleted value.

```
>>> del L1 [4]
>>> print L1
```

del () with slicing

Consider the following example:

Examples

```
L1 = [1, 2, 5, 4, 70, 10, 90, 80, 50]
>>> del L1 [2:4]
>>>print L1
[1, 2, 70, 10, 90, 80, 50]
```

will remove 2nd and 3rd element from the list. As we know that slice selects all the elements up to 2nd index but not the 2nd index element. So **4th element** will remain in the
list.

```
>>> L1[2:4] = [ ]
```

Will delete the slice

```
>>>print L1

[1, 2, 70, 10, 90, 80, 50]
```

Note:

For arranging elements in an order Python provides a method **sort ( )** and a function **sorted ( )**. sort ( ) modifies the list in place and sorted ( ) returns a new sorted list.

Its Syntax are:
sort ([cmp [, key [, reverse]]])
sorted (list [, cmp [, key [, reverse]]])

```
L1 = [1, 2, 5, 4, 70, 10, 90, 80, 50]

L2=sorted(L1)

print(L1)

print(L2)
```

Output

```
[1, 2, 5, 4, 70, 10, 90, 80, 50]
[1, 2, 4, 5, 10, 50, 70, 80, 90]
```

```
L1 = ['chennai','delhi','mumbai']
L1.sort(key=len)
print(L1)
['delhi', 'mumbai', 'chennai']
```

Matrix implementation using list

Its syntax is:
a=[[0 for col in range(number of cols)]for row in range(number of rows)]

```
m=int(input("how many rows"))
n=int(input("how many cols"))
ar=[[0 for col in range(n)]for row in range(m)]
for i in range(m):
    for j in range(n):
        ar[i][j]=int(input("enter a no"))
for i in range(m):
    for j in range(n):
        print(ar[i][j],end="\t")
    print()
```

Tuples

A tuple is a collection of Python immutable objects separated by commas. Tuples are represented by parentheses (). Like lists, tuples also work well with basic operations as shown in the table given below.

Python Expression	Results	Description
len((1,2))	2	Length
(1,2)+(4,5)	(1,2,4,5)	Concatenation

(‘CS’,)*2	(‘CS’,’CS’)	Repetition
5 in (1,2,3)	False	Membership

for x in (4,2,3):
print(x,end=’ ‘)
	4 2 3	Iteration

Tuple Creation

If we need to create a tuple with a single element, we need to include a final comma.

Example

```
>>> t=10,
>>> print t
(10,)
```

Another way of creating tuple is built-in function tuple ().

Syntax:

T = tuple()

Example

```
>>> T=tuple()

>>> print T
()
```

Add new element to Tuple

We can add new element to tuple using + operator.

Example

```
>>> t=(10,20,30,40)
>>> t+(60,) # this will not create modification of t.
(10, 20, 30, 40, 60)
>>> print t
(10, 20, 30, 40)
>>> t=t+(60,) # this will do modification of t.
>>> print t
(10, 20, 30, 40, 60)
```

Example

Write a program to input ‘n’ numbers and store it in tuple.

Code

```
t=tuple()
n=input("Enter any number")
print " enter all numbers one after other"
for i in range(n):
    a=input("enter number")
    t=t+(a,)
print "output is"
print t
```

```
>>> T=tuple() #create empty tuple
>>> print T
()
>>> l=list(T) #convert tuple into list
>>> l.append(10) #Add new elements to list
>>> l.append(20)
>>> T=tuple(l) #convert list into tuple
>>> print T
(10, 20)
```

Tuple assignment

Example

```
>>> T1=(10,20,30)
>>> T2=(100,200,300,400)
>>> print T1
(10, 20, 30)
>>> print T2
(100, 200, 300, 400)
>>> T1,T2=T2,T1 # swap T1 and T2
>>> print T1
(100, 200, 300, 400)
>>> print T2

(10, 20, 30)
```

Tuple slicing

Syntax

```
Tuplename[start:stop:step]
```

Example

```
>>> T=(10,20,30,40,50)
>>> T1=T[2:4]
>>> print T1
(30, 40)

>>> T[:]
(10, 20, 30, 40, 50)
```

Will produce a copy of the whole tuple.

```
>>> T[::2]
(10, 30, 50)
```

```
>>> T[:3]
(10, 20, 30)
```

Will produce 0 to 2(3-1)

```
>>> T[2:]
(30, 40, 50)
```

Will produce from 2 to end.

TUPLE FUNCTIONS

Function	Description	Example

cmp(tuple1,tuple2)	Compares element from both the tuples. This is used to check whether the given tuples are same or not. If both are same, it will return ‘zero’, otherwise return 1 or -1. If the first tuple is big, then it will return 1, otherwise return -1.	>>> T1=(10,20,30)
		>>> T2=(100,200,300)
		>>> T3=(10,20,30)
		>>> cmp(T1,T2)
		-1
		>>> cmp(T1,T3)
		0
		>>> cmp(T2,T1)
		1

len(tuple)	Returns the total length of the tuple	>>> T2=(100,200,300,400,500)
		>>> len(T2)
		5

max(tuple)	Returns the item from the tuple with the maximum value	>>> T=(100,200,300,400,500)
		>>> max(T)
		500

min(tuple)	Returns the item from the tuple with the minimum value	>>> T=(100,200,300,400,500)
		>>> min(T)
		100

tuple(seq)	Converts a list into tuple	

Dictionaries

A dictionary is like a list except that in a list we have to access it using an index, whereas items in a dictionary can be accessed using a unique key, which can be a number, string or a tuple. The items in a dictionary can be changed but keys are an immutable data type. Each key is separated from its value by a colon(:) , the items are separated by commas, and the entire elements (key-value pair) are enclosed in curly braces {}

Syntax

```
  Dictname={‘key1’:’value1’,’key2’:’value2’,……}
```

```
>>> A={1:"one",2:"two",3:"three"}
>>> print A
{1: 'one', 2: 'two', 3: 'three'}
```

Creation, initializing and accessing the elements in a Dictionary

The function dict ( ) is used to create a new dictionary with no items. This function is called built-in function. We can also create dictionary using {}.

```
>>> D=dict()
>>> print D
{}
```

{} represents empty string. To add an item to the dictionary (empty string), we can use square brackets for accessing and initializing dictionary values.

Example

```
>>> H=dict()
>>> H["one"]="keyboard"
>>> H["two"]="Mouse"
>>> H["three"]="printer"
>>> H["Four"]="scanner"
>>> print H
{'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}
```

Updating dictionary elements

Syntax

```
   Dictname[key]=value
```

Built In Dictionary Functions

Function	Description	Example

cmp(dict1, dict2)	This is used to check whether the given dictionaries are same or not. If both are same, it will return ‘zero’, otherwise return 1 or -1. If the first dictionary having more number of items, then it will return 1, otherwise return -1.	>>> D1={'sun':'Sunday','mon':'Monday','tue':'Tuesday','wed':'Wednesday','thu':'Thursday','fri':'Friday','sat':'Saturday'}
		>>> D2={'sun':'Sunday','mon':'Monday','tue':'Tuesday','wed':'Wednesday','thu':'Thursday','fri':'Friday','sat':'Saturday'}
		>>> D3={'mon':'Monday','tue':'Tuesday','wed':'Wednesday'}
		>>> cmp(D1,D3) #both are not equal
		1
		>>> cmp(D1,D2) #both are equal
		0
		>>> cmp(D3,D1)
		-1

len(dict)	Returns the total items present in the dictionary	>>> H={'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}
		>>> len(H)
		4

str(dict)	Produces a printable string representation of a dictionary	H={'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}
		st=str(H)
		
		print(st,len(st))
		{'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'} 74
		
		print(H,len(H))

		{'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'} 4
type(var)	Returns the type of the variable passed as argument	

METHODS

Method	Description	Example

dict.clear()	Removes all element from dictionary	>>> D={'mon':'Monday','tue':'Tuesday','wed':'Wednesday'}
		>>> print D
		{'wed': 'Wednesday', 'mon': 'Monday', 'tue': 'Tuesday'}
		>>> D.clear( )
		>>> print D
		{}

dict.copy()	Returns a shallow copy of dictionary	H={'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}
		K=H.copy()
		print(H)
		print(K)
		K['three']='LED'
		print(H)
		print(K)
		{'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}
		{'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}
		{'Four': 'scanner', 'three': 'printer', 'two': 'Mouse', 'one': 'keyboard'}
		{'Four': 'scanner', 'three': 'LED', 'two': 'Mouse', 'one': 'keyboard'}

dict.items()	Returns a list of dict’s (key,value) tuple pairs	D={'sun':'Sunday','mon':'Monday','tue':'Tuesday','wed':'Wednesday','thu':'Thursda
		y','fri':'Friday','sat':'Saturday'}
		>>> D.items()

		[('wed', 'Wednesday'), ('sun', 'Sunday'), ('thu', 'Thursday'), ('tue', 'Tuesday'), ('mon',
		'Monday'), ('fri', 'Friday'), ('sat', 'Saturday')]
		>>>print(D.items())
		dict_items([('sun', 'Sunday'), ('mon', 'Monday'), ('tue', 'Tuesday'), ('wed', 'Wednesday'), ('thu', 'Thursday'), ('fri', 'Friday'), ('sat', 'Saturday')])

dict.keys()	Returns a list of dictionary keys	D={'sun':'Sunday','mon':'Monday','tue':'Tuesday','wed':'Wednesday','thu':'Thursday','fri':'Friday','sat':'Saturday'}
		print(D.keys())
		dict_keys(['sun', 'mon', 'tue', 'wed', 'thu', 'fri', 'sat'])

dict.get(key,x)	There are two arguments (k, x) passed in ‘get( )’ method. The first argument is key value, while the second argument is corresponding value. If a dictionary has a given key (k), which is equal to given value	>>>
		D={'sun':'Sunday','mon':'Monday','tue':'Tuesday','wed':'Wednesday','thu':'Thursda
		y','fri':'Friday','sat':'Saturday'}
		>>> D.get('wed',"wednesday") # corresponding value wed
		'Wednesday'
		>>> D.get("fri","monday") # default value of fri
		'Friday'
		>>> D.get("mon") # default value of mon
		'Monday'
		>>> D.get("ttu") # None
		D={'sun':'Sunday','mon':'Monday','tue':'Tuesday','wed':'Wednesday','thu':'Thursday','fri':'Friday'}
		print(D.get('sat','Day'))
		Day

	(x), it returns the corresponding value (x) of given key (k). However, if the dictionary has no key-value pair for given key (k), this method returns the default values same as given key value. The second argument is optional. If omitted and the dictionary has no key equal to the given key value, then it returns None.	

dict.setdefault(key,default=None)	Similar to get(), but will set dict[key]= default if	D={'sun':'Sunday','mon':'Monday','tue':'Tuesday','wed':'Wednesday','thu':'Thursday','fri':'Friday'}
		D.setdefault('sat','Day')
		print(D)

	key is not already in dict	{'sun': 'Sunday', 'mon': 'Monday', 'tue': 'Tuesday', 'wed': 'Wednesday', 'thu': 'Thursday', 'fri': 'Friday', 'sat': 'Day'}

dict1.update(dict2)	Adds dict2 key-value pairs to dict1	>>> d1={1:10,2:20,3:30}
		>>> d2={4:40,5:50}
		>>> d1.update(d2)
		>>> print d1
		{1: 10, 2: 20, 3: 30, 4: 40, 5: 50}
		{1: 10, 2: 30, 3: 30, 5: 40, 6: 60} # k>>> d1={1:10,2:20,3:30} # key 2 value is 20
		>>> d2={2:30,5:40,6:60} #key 2 value is 30
		>>> d1.update(d2)
		>>> print d1
		{1: 10, 2: 30, 3: 30, 5: 40, 6: 60}

dict.values()	Returns a list of dict values	D={'sun':'Sunday','mon':'Monday','tue':'Tuesday','wed':'Wednesday','thu':'Thursday','fri':'Friday','sat':'Saturday'} >>> D.values() ['Wednesday', 'Sunday', 'Thursday', 'Tuesday', 'Monday', 'Friday', 'Saturday']

	PYTHON REVISION TOUR	
1.	State True or False “Variable declaration is implicit in Python.” **Ans : Yes**	1
2.	Which of the following is an invalid datatype in Python? (a) Set (b) None(c)Integer (d)Real **Ans : None**	1
3.	Given the following dictionaries dict_exam={"Exam":"AISSCE", "Year":2023} dict_result={"Total":500, "Pass_Marks":165} Which statement will merge the contents of both dictionaries? a. dict_exam.update(dict_result) b. dict_exam + dict_result c. dict_exam.add(dict_result) d. dict_exam.merge(dict_result) Ans : a	1
4.	Consider the given expression: not True and False or True Which of the following will be correct output if the given expression is evaluated? F and false or true (a) True (b) False (c) NONE (d) NULL Ans a	1
5.	Select the correct output of the code: a = "Year 2022 at All the best"	1

	a = a.split('2') b = a[0] + ". " + a[1] + ". " + a[3] print (b) (a) Year . 0. at All the best (b) Year 0. at All the best (c) Year . 022. at All the best (d) Year . 0. at all the best Ans Year . 0. at All the best	

6.	Which of the following statement(s) would give an error after executing the following code?
	
	S="Welcome to class XII" # Statement 1
	print(S) # Statement 2
	S="Thank you" # Statement 3
	S[0]= '@' # Statement 4
	S=S+"Thank you" # Statement 5
	
	(a) Statement 3
	(b) Statement 4
	(c) Statement 5
	(d) Statement 4 and 5
	Ans b
		1

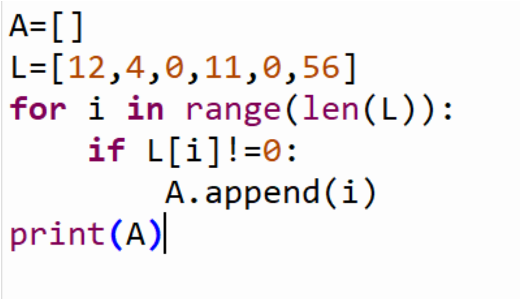
7.	What will the following expression be evaluated to in Python? print(15.0 / 4 + (8 + 3.0)) (a) 14.75 (b)14.0 (c) 15 (d) 15.5 Ans 14.75	1
8.	(a) Given is a Python string declaration: myexam="@@CBSE Examination 2022@@" Write the output of: print(myexam[::-2]) Ans: @20 otnmx SC@ (b) Write the output of the code given below: my_dict = {"name": "Aman", "age": 26} my_dict['age'] = 27 my_dict['address'] = "Delhi" print(my_dict.items()) Ans dict_items([('name', 'Aman'), ('age', 27), ('address', 'Delhi')])	1 1

9. Write a program which stores the a list and create another list which contains the indexes of all non zero elements. 3

For example:

If L contains [12,4,0,11,0,56]

The index List will have - [0,1,3,5]



```
A=[]
L=[12,4,0,11,0,56]
for i in range(len(L)):
    if L[i]!=0:
        A.append(i)
print(A)
```

10 (a) Predict the output of the code given below:

```
s="welcome2cs"
n = len(s)
m=""
for i in range(0, n):
    if (s[i] >= 'a' and s[i] <= 'm'):
        m =m +s[i].upper()
    elif (s[i] >= 'n' and s[i] <= 'z'):
        m = m +s[i-1]
    elif (s[i].isupper()):
        m = m + s[i].lower()
    else:
        m = m +'&'
print(m)
```

sELCcME&Cc

11 State True or False: 1

“In a Python program, if a break statement is given in a nested loop, it terminates the execution of all loops in one go.”

No. It terminates rhe inner loop.

12	In a table in MYSQL database, an attribute A of datatype varchar(20)has the value “Keshav”. The attribute B of datatype char(20) has value “Meenakshi”. How many characters are occupied by attribute A and attribute B? a. 20,6 b. 6,20 c. 9,6 d. 6,9 Ans : b	1
13	What will be the output of the following statement: print(3-2**2**3+99/11) a. 244 b. 244.0 c. -244.0 d. Error c	1
14	Which of the following will delete key-value pair for key = “Red” from a dictionary D1? a. delete D1("Red") b. del D1["Red"] c. del.D1["Red"] d. D1.del["Red"] B	1

15	Consider the statements given below and then choose the correct output from the given options:
	pride="#G20 Presidency"
	print(pride[-2:2:-2])
		1

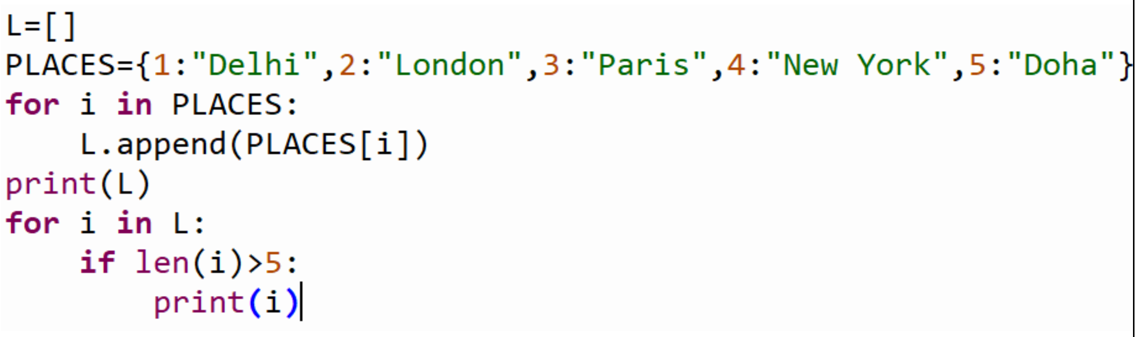
	Options a. ndsr b. ceieP0 c. ceieP d. yndsr ceieP0	

16	Which of the following statement(s) would give an error during execution of the following code?
	tup = (20,30,40,50,80,79)
	print(tup) #Statement 1
	print(tup[3]+50) #Statement 2
	print(max(tup)) #Statement 3
	tup[4]=80 #Statement 4
	Options:
	a. Statement 1
	b. Statement 2
	c. Statement 3
	d. Statement 4
	Statement 4
		1

17	Write a program in Python, to create the dictionary to store numbers as keys and values as cities and only displays cities whose length is more than 5. For example, Consider the following dictionary PLACES={1:"Delhi",2:"London",3:"Paris",4:"New York",5:"Doha"} The output should be:	2

LONDON

NEW YORK

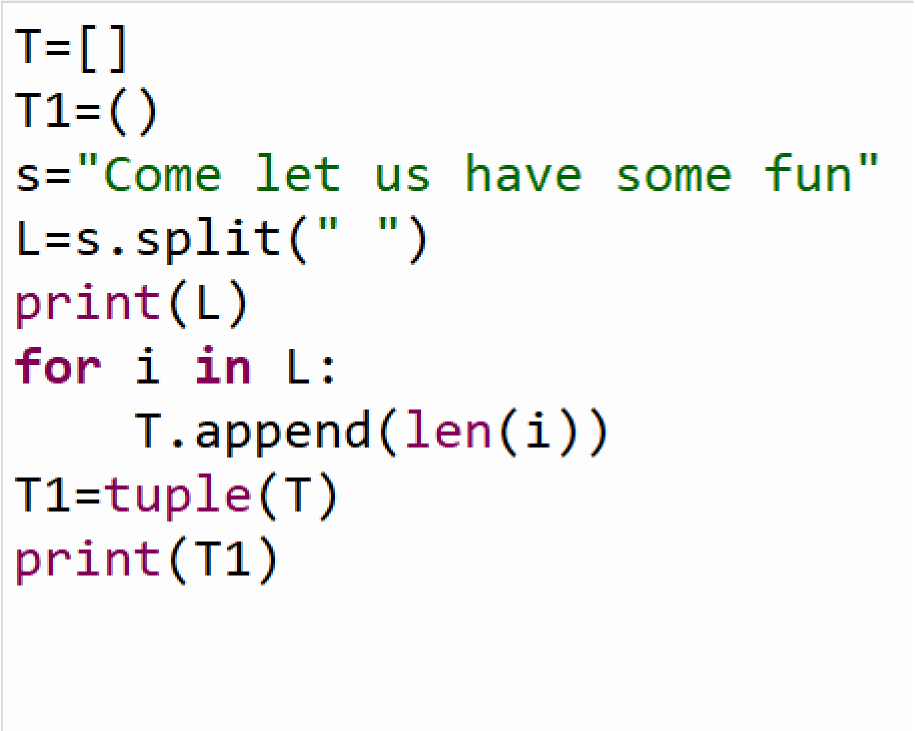


```
L=[]
PLACES={1:"Delhi",2:"London",3:"Paris",4:"New York",5:"Doha"}
for i in PLACES:
    L.append(PLACES[i])
print(L)
for i in L:
    if len(i)>5:
        print(i)
```

OR

Write a program to accept a string and returns a tuple containing length of each word of a string. For example, if the string is "Come let us have some fun", the

tuple will have (4, 3, 2, 4, 4, 3)



```
T=[]
T1=()
s="Come let us have some fun"
L=s.split(" ")
print(L)
for i in L:
    T.append(len(i))
T1=tuple(T)
print(T1)
```

18 Predict the output of the following code: 2

```
S = "LOST"
L = [10,21,33,4]
D={}
for I in range(len(S)):
    if I%2==0:
        D[L.pop()] = S[I]
    else:
        D[L.pop()] = I+3

for K,V in D.items():
    print(K,V,sep="*")
```

19 Write the Python statement for each of the following tasks using BUILT- IN functions/methods only: 1+1= 2

(i) To insert an element 200 at the third position, in the list L1.
L1.insert(200,2)

(ii) To check whether a string named, message ends with a full stop / period or not.
if S[-1]=='.':

20 State whether the statement is True or False?
No matter the underlying data type, if values are equal returns true.

True

21 Every variable in Python holds an instance of an object. These Objects are of two types.
(a) Mutable (b) immutable (c) both a and b
(d) instance

C

22 T=1,2,3
(a) Tuple (b) List (c) variable (d) None of these
A

23 Evaluate the following

Not 20>21 and 6>5 or 7>9

(a) True (b) False (c) 3 (d) None of these

True

24 What will be the output of the following Python code snippet?

```
X=11
if X<=10:
    X+=10
if X<=20:
    X+=10
if X<=30:
    X+=30
print(X)
```

a. 21

b. 31

c. 61

d. None of these

d.

25 Select the output

```
X="abcdef"
i="a"
while( I in X):
    print(I,end=’’)
```

(a) Error

(b) a

(c) infinite loop

(d) No ouput

c- Infinite loop

26 Give output

```
RS=''
S='important'
for i in S:
    RS=i+RS
print(RS)
```

```
tnatropmi
```

27 Predict the output

```
import random
x=random.random( )
y=random.randint(0,4)
print(int(x),":",y+int(x))
```

28 The________statement is an empty statement in Python.

Pass

29 Which of the following is not a keyword?

(a) eval (b) assert (c) nonlocal (d) pass

Nonlocal

30 What will be the output for the

following Python statements? D=

{“Amit”:90, “Reshma”:96, “Suhail”:92,

“John”:95}

print(“John” in D, 90 in D, sep= “#”)

(a) True#False (b)True#True (c) False#True (d) False#False

Ans A

31 What will the following Python

statement evaluate to?

print (5 + 3 ** 2 / 2)

(a) 32 (b) 8.0 (c) 9.5 (d) 32.0

Ans : 9.5

32 Consider the list aList=[“SIPO”, [1,3,5,7]].
What would the following code print?

```
print(alist[0],aList[1][1])
```

(a) S, 3 (b) S, 1 (c) I, 3 (d) I, 1

Ans – a

33 Identify the errors in the following code:

```
MyTuple1=(1, 2, 3)    #Statement1

MyTuple2=(4)           #Statement2

MyTuple1.append(4)    #Statement3

print(MyTuple1, MyTuple2)

#Statement4
```

(a) Statement 1 (b) Statement 2 (c) Statement 3 (d) Statement 2 &3

Ans Statement3

34 Suppose str1= ‘welcome’. What will the following expressions print

(a) str[: : -6]

(b) str[: : -1][: : -6]

(c) str[: :6]

(d) str[0] + str[-1]

a) Ew
b) We
c) We
d) We

35 Give the output

```
d1={'a':10, 'b':2, 'c':3}

str1= ' '

for i in d1:

    str1=str1+str(d1[i])+ ' '

    str2=str1[ : -1]
```

2

```
print(str2[: : -1])
```

Ans 3 2 01

36 What possible output(s) are expected to be displayed on screen at the time of execution of the following code? Also specify the maximum and minimum value that can be assigned to variable X.

```
import random
L=[10,7,21]
X=random.randint(1,2)
for i in range(X):
   Y=random.randint(1,X)
```

(a) 10 $ 7 $ (b) 21 $ 7 $ (c) 21 $ 10 $

(d) 7 $

Ans d

37 Write a function INDEX_LIST(L), where L is the list of elements passed as argument to the function. The function returns another list named 'indexList' that stores the indices of all Non-Zero Elements of L. For example: If L contains [12,4,0,11,0,56] The indexList will have - [0,1,3,5]

38 Predict the output of the following Python code:

```
tup1 = ("George","Anderson","Mike","Luke","Amanda")
list1 =list(tup1)
list2 = []
for i in list1:
    if i[-1]=="e":
        list2.append(i)
tup2 = tuple(list2)
print(tup2)
```

Ans

```
('George', 'Mike', 'Luke')
```

39 Write a program in python, which accepts a list of numbers and a numeric value by which all elements of the list are shifted to left.

Sample Input data of the

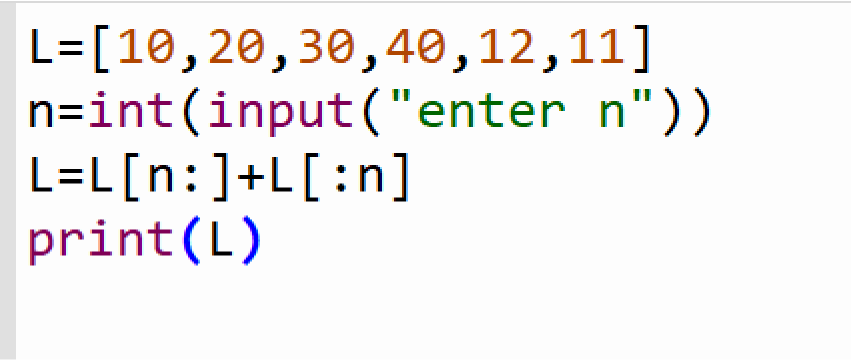
list

Arr=[10,20,30,40,12,11] and

n=2

Output

Arr :[30,40,12,11,10,20]



```
L=[10,20,30,40,12,11]
n=int(input("enter n"))
L=L[n:]+L[:n]
print(L)
```

40

```
s = 'mAhAtMaGaNdHi'

i = 0

while i<len(s):
    if (s[i].islower()):

print(s[i].upper(),end='
')
    if (s[i].isupper()):

print(s[i].lower(),end='
')
    i += 1



   M a H a T m A g A n h I
```

Select the correct output of the following code:

```
>>>str1 = ‘India is a Great Country’
>>>str1.split(‘a’)
```

e) [‘India’, ’is’, ’a’,’ Great’, ’Country’]
f) [‘India’, ’is’, ’Great’, ’Country’]
g) [‘Indi’, ’is’, ’Gre’, ’t Country’]
h) [‘Indi’, ’is’, ’Gre’, ’t’, ‘Country’]

Ans. c) [‘Indi’, ’is’, ’Gre’, ’t Country’]

41 Which of the following will be the output of the code:

```
mySubject = “Computer Science”
print(mySubject[:3] +
mySubject[3:])
```

e) Com
f) puter Science
g) Computer Science
h) Science Computer

	Ans. c) Computer Science	
42	How will the following expression be evaluated in Python? 2 + 9 * ((3 * 12) - 8) / 10 e) 29.2 f) 25.2 g) 27.2 h) 27 **Ans. c) 27.2**	
43	(c) Given is a Python string declaration: str1="!!Welcome to Python!!" Write the output of: print(str1[::-2]) **Ans.** **!nhy teolW!** (b) Write the output of the code given below: dict1 = {"name": "Suman", "age": 36} dict1['age'] = 27	

```
dict1['address'] = "Chennai"
print(dict1.keys())
```

Ans. **dict_keys(['name', 'age', 'address'])**

44 Predict the output of the following Python code:

```
tup1 =
("George","Anderson","Mike","Luke","Amand
a") list1 =list(tup1)
list2 = []
for i in list1:
if i[-1]=="e":
  list2.append(i)
tup2 = tuple(list2)
print(tup2)
```

Ans. ('George', 'Mike', 'Luke')

45 Write a function LShift(arr,n) in python, which accepts a list of numbers and a numeric value by which all elements of the list are shifted to left.

Sample Input data of the

list

Arr=[10,20,30,40,12,11] and

n=2

```
Output
Arr :[30,40,50,12,11,10,20]
```

Ans.

```
def LShift(Arr,n):
    L=len(Arr) for x
  in range(0,n):
        y=Arr[0]
        for i in range(0,L-1):
            Arr[i]=Arr[i+1]
        Arr[L-1]=y
    print(Arr)
```

46 b) Predict the output of the code given below:

```
s = 'MoHaNdAs GaNdHi'
 i = 0
while i<len(s):
  if (s[i].islower()):
    print(s[i].upper(),end=' ')
  if (s[i].isupper()):
    print(s[i].lower(),end=
  ' ') i += 1
```

```
Ans. m O h A n D a S g A n D h I
```

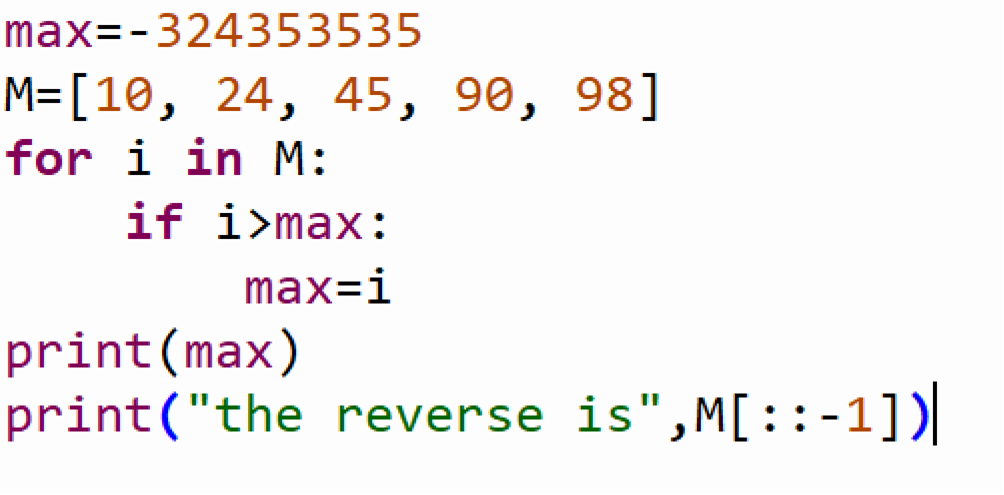
47 Write a python program to find the largest element in a list and then reverse the list contents and display it. Don’t use in-built functions for the program.

E.g.

M = [10, 24, 45, 90, 98]

Output

Largest in the give n List
is 98 Reverse list:
[98,90,45,24,10]



```
max=-324353535
M=[10, 24, 45, 90, 98]
for i in M:
    if i>max:
        max=i
print(max)
print("the reverse is",M[::-1])
```

48
Which of the following is not a valid identifier in Python?

a) KV2

b) _main

c) Hello_Dear1

d) 7 Sisters

49
A variable created or defined in a function body is known as…

a) local
b) global
c) built-in
d) instance

50

Suppose

list1 = [0.5 * x for x in

range(0,4)],

list1 is

a) [0, 1, 2, 3]

b) [0, 1, 2, 3, 4]

c) [0.0, 0.5, 1.0, 1.5]
d) [0.0, 0.5, 1.0, 1.5, 2.0]

Ans : c

50
Which statement is not correct

a)The statement x = x + 10 is a valid statement

b)List slice is a list itself.

c)Lists are immutable while strings are mutable.

d)Lists and strings in pythons support two way indexing.

51
What will be the output of following code snippet:
msg = “Hello Friends”
msg [: : -1]

Ans
sdneirF olleH

52
Suppose a tuple T1 is declared as
T1 = (10, 20, 30, 40, 50)
which of the following is incorrect?
a)print(T[1])
b) T[2] = -29
c) print(max(T))
d) print(len(T))
Ans : b

53

Rewrite the following code after removing the syntactical error(if any).Underline each correction:

```
X=input("Enter a Number")
if x % 2 =0:
    for i range (2*x):
        print i
    loop else:
        print "#"
```

54.	“break and continue statements are conditional statements”. True orFalse ?	1
55	Which of the following are not valid keywords? (a)False (b)Math (c)WHILE (d)break Ans : d	1
56	Consider two lists L1= [2,4,6] and L2=[6,7,8] What will be the output after the following code is executed? L1=[2,4,6] L2=[6,7,8] L1 = L2 L2.append(5) print(L1) a. [2,4,6,5] b. [6,7,8,5] c. [6,7,8] e. [2,4,6] Ans : b	1
57	Which of the following are valid identifiers ? a. price2 b. %sales% c) Else d. unit price Ans a,c	1

58 Look at the following and determine the output . 1

```
 S="Work hard"
 print(S[1:-1])
```

a. ‘rah dro’
b. ‘work har’
c. 'ork har'
d. ‘drah droW’
Ans : c

59. What will be the output of the following expression? 1

```
print(  95//3**2**2+12-3%2)
```

a. 12
b. 14
c. 11
d. 10

Ans : 12

60 Rewrite the following code after removing syntax error:

```
Runs = ( 10, 5, 0, 2, 4, 3 )
for I in Runs:
    if I=0:           # ==
        print(Maiden Over)
     else:
        print(Not Maiden)   “ Double quote missing”
```

a) What will be the output of the following python code?

```
S="UVW"
L=[10,20,30]
D={ }
N=len(S)
for I in range(N):
    D[L[I]]=S[I]
for K,V in D.items():
    print(K, V, sep='*', end = ', ')
```

Ans :
10*U, 20*V,30*W

b) Predict the output of the Python code given below:

```
t1=(10,20,"book",30,9.5,"item",[12,13],(3,4),30,5,30)
print( t1.index(30)*t1.count(30))
print(t1[-8:-4])
```

Ans :
9
(30,9.5,”item”,[12,13])

What are the possible outputs and why

```
import random
ar = [2, 3, 4, 5, 6, 7]
minn = random.randint (1, 3)
maxn = random.randint (2, 4)
for i in range (minn, maxn + 1):
     print (ar [i], end = ‘#’)
```

(a) 3# 4# 5# (b) 5# 6# 7# (c) 1# 4# 7# (d) 4# 5# 7#
Ans : a

1.	State True or False: “All keywords in Python are in lowercase.”	1
2.	Identify the invalid identifier. a) keyword b) token c) operator d) and Ans d	1
3.	Consider the following Dictionary: D = {1: ['Amit',23,21], 2: ['Suman',45,34], 3: 'Ravi', 4: 'Anuj'} m = D.get(2) print(m[2]) Write the output of the given code: a) 34 b) 45 c) m d) Suman Ans :	1
4.	Consider the given expression: 12 % 4 + 6 + 4 / / 3 Which of the following will be correct output if the given expression is evaluated? a) 7.0 b) 7.33 c) 7 d) 7.03 Ans : c	1

	Select the correct output of the code:
	>>> s="i love my country"
	>>> r = "i love my class"
	>>> s[2:6] + r[-7:]
	a) 'loveyclass'
	b) 'lovey class'
	c) 'loveclass'
	d)'love class'
	Ans : b	

Consider the following
code:

```
tup = (20, 30, 40, 50,
    80, 79)
    print(tup)            #Statement 1
    print(tup[3]+50)      #Statement 2
    print(max(tup))       #Statement 3
    tup[4]=80             #Statement 4
```

Which of the following statement(s) would give an error after executing the following code?

(a) Statement 1

(b) Statement 2

(c) Statement 3

Statement 4

Given is a Python String
declaration:

```
exam="CBSE Examination
2023"
```

Write the output of:

```
print(exam[ - 10 : - 2 : 2 ])
```

Ans : ain2

d. Write the output of the code given below:

```
    A = {1: "One", 2: "Two", 3: "Three", 4:"Four"}
    B = {1: 'Amit', 2: 'Sunil', 5: 'Lata', 6: 'Suman'}
    A.update (B)
   print(A)
```

Ans:

{1: 'Amit', 2: 'Sunil', 3: 'Three', 4: 'Four', 5: 'Lata', 6: 'Suman'}

```
str='Exam 2023'
s='#'
nwstr=' '
for x in str:
   if x.isdigit():
      nwstr=nwstr + s
   elif x.isalpha():
      if x.isupper():
         nwstr=nwstr+x.lower()
      else:
         nwstr=nwstr+x.upper()
   else:
      nwstr=nwstr + '*'
print(nwstr)
```

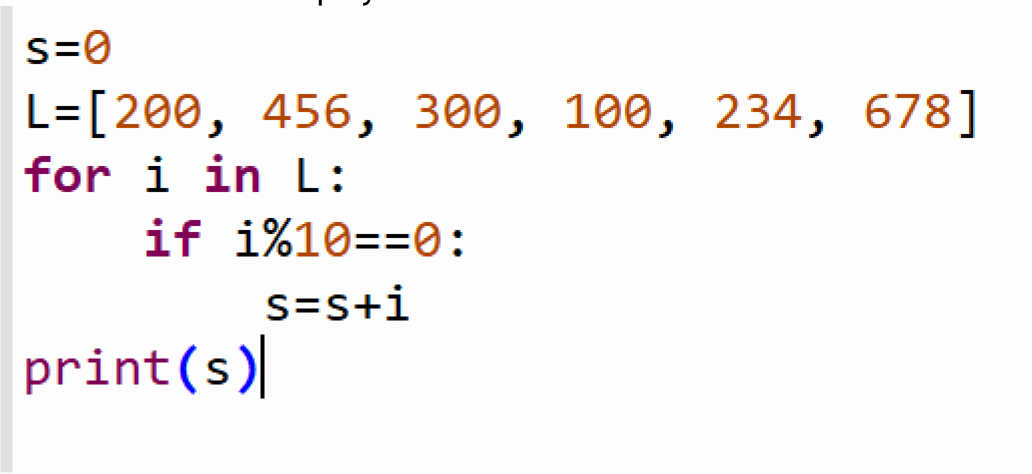
Ans: eXAM*####

Write a program that prints the sum of all the values in the List of SCORES which are ending with the number zero(0).

For example:

If the SCORE contain [200, 456, 300, 100, 234, 678]

The sum should be displayed as 600



```
s=0
L=[200, 456, 300, 100, 234, 678]
for i in L:
    if i%10==0:
        s=s+i
print(s)
```

a Predict the output of the code given below:

```
s="Rs.12"
n, m = len(s), " "
for i in range(0, n):
   if s[i].islower():
      m = m +s[i]
   elif s[i].isupper():
      m = m +s[i+1]
   elif s[i].isdigit():
      m = m*int(s[i])
   else:
      m = '@'+m
print(m)
```

@ss@ss

[1]