1.what is list and tuple?

List is a data type and mutable.

lis=[1,2,3]

Tuple is a datatype and immutable.

tup=(1,2,3)

2.when to use list and when to use tuple

List is used when you want to append a value to any list.

Tuple is used when appending is not neccessary.

3.what is instance?,where can you use instance?

instance is used to check the data type of a given value.

eg:isinstance('string1',str)

4.basic data types in python

Basic data types of the python:-

1) Number

2) string

In python number will support any type of number like real,floatinng point etc.

Example: var=10

var=10.5

In python we can define the string in single , double and triple qutation.

Example :- Mystr= 'HELLO INDIA'

Mystr= "HELLO INDIA"

Mystr= """HELLO INDIA""".

5.what are different data structures in python?

Basic data structures in python are listed below

1) List

2) Tuple

3) Dictionary

4) set

5) frozonset

1)List is basic datatype of python.List is a mutable types.we can change the value of element in the list.

Example:- mylist=[1,2,3,4,"ABC"]

2)Tuple is like a list but we can not change the value of element because tuple is a imutable datatypes .

Example:-mytuple=(1,2,3,4,"ABC").

3)Dictionary is a key value pair.

Example:-mydict={'Name':"RAHUL",'EMP\_ID':839}.

4) set is remove the duplicates .

Example :- mylist=[1,1,1,1,0,3,2,0,4]

res=set(mylist)

6.how can u add a list(list of elements) to an existing list?

list1=[1,2,3]

list2=[a,b,c]

res=list1.extend(list2)

7.what is set?

Set is predefine keyword in python using this we can remove the duplicate and sort the data in assending order.

Example :- mylist=[1,1,1,1,0,3,2,0,4]

res=set(mylist)

//Ouyput : [0,1,2,3,4]

8.what is frozonset?

Sets are mutable, and may therefore not be used, for example, as keys in dictionaries.

Another problem is that sets themselves may only contain immutable (hashable) values, and thus may not contain other sets.

Because sets of sets often occur in practice, there is the frozenset type, which represents immutable (and, therefore, hashable) sets.

>>> a = frozenset([1, 2, 3])

>>> b = frozenset([2, 3, 4])

>>> a.union(b)

frozenset([1, 2, 3, 4])

9.what different keywords in python,can we use keywords as variables?

$ python

>>> import keyword

>>> print keyword.kwlist

['and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else',

'except', 'exec', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is',

'lambda', 'not', 'or', 'pass', 'print', 'raise', 'return', 'try', 'while', 'with',

'yield']

we can't use keywords as variables.

10.print the elements of the list by skipping 2 elements ?

ex l=[1,2,3,4,5,6,7,8,9] output = [1,4,7]

print l[::3]

11.Reverse a list with out using built in functions?

list=[1,2,3,4,5]

print list[::]

or

def reverse(data\_list):

length = len(data\_list)

s = length

new\_list = [None]\*length

for item in data\_list:

s = s - 1

new\_list[s] = item

return new\_list

list1=[1,2,3,4,5]

list2=reverse(list1)

print list2

12.what are dict? write dict functions?

dict :

Each key is separated from its value by a colon (:), the items are separated by commas, and the whole thing is enclosed in curly braces.

An empty dictionary without any items is written with just two curly braces, like this: {}.

Keys are unique within a dictionary while values may not be. The values of a dictionary can be of any type, but the keys must be of an immutable data type such as strings, numbers, or tuples.

dict = {'Name': 'Zara', 'Age': 7, 'Name': 'Manni'}

print ("dict['Name']: ", dict['Name'])

dict functions

1.cmp(dict1, dict2)

#!/usr/bin/python

dict1 = {'Name': 'Zara', 'Age': 7};

dict2 = {'Name': 'Mahnaz', 'Age': 27};

dict3 = {'Name': 'Abid', 'Age': 27};

dict4 = {'Name': 'Zara', 'Age': 7};

print "Return Value : %d" % cmp (dict1, dict2)

print "Return Value : %d" % cmp (dict2, dict3)

print "Return Value : %d" % cmp (dict1, dict4)

o/p:

Return Value : -1

Return Value : 1

Return Value : 0

2.len(dict)

#!/usr/bin/python

dict = {'Name': 'Zara', 'Age': 7};

print "Length : %d" % len (dict)

o/p: Length : 2

3.str(dict)

dict = {'Name': 'Zara', 'Age': 7};

print "Equivalent String : %s" % str (dict)

o/p: Equivalent String : {'Age': 7, 'Name': 'Zara'}

4.type(dict)

dict = {'Name': 'Zara', 'Age': 7};

print "Variable Type : %s" % type (dict)

O/P: Variable Type : <type 'dict'>

13.find second largest number in the given list without using built in functions?

>>> numbers = [20,67,3,2.6,7,74,2.8,90.8,52.8,4,3,2,5,7]

>>> if numbers[0]>numbers[1]):

... m, m2 = numbers[0], numbers[1]

... else:

... m, m2 = numbers[1], numbers[0]

...

>>> for x in numbers[2:]:

... if x>m2:

... if x>m:

... m2, m = m, x

... else:

... m2 = x

...

>>> m2

74

14.Transform list to other data structure?

>>> a = [66.25, 333, 333, 1, 1234.5]

>>> print a.count(333), a.count(66.25), a.count('x')

2 1 0

>>> a.insert(2, -1)

>>> a.append(333)

>>> a

[66.25, 333, -1, 333, 1, 1234.5, 333]

>>> a.index(333)

1

>>> a.remove(333)

>>> a

[66.25, -1, 333, 1, 1234.5, 333]

>>> a.reverse()

>>> a

[333, 1234.5, 1, 333, -1, 66.25]

>>> a.sort()

>>> a

[-1, 1, 66.25, 333, 333, 1234.5]

>>> a.pop(1234.5)

>>> a

[-1, 1, 66.25, 333, 333]

15.how to create flat list from list of lists? ex l=[[1,2],[6,3],[4,5]] o/p = [1,2,6,3,4,5]

>>> import operator

>>> l = [[1,2],[6,3],[4,5]]

>>> reduce(operator.concat, l)

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

or

>>> l = [[1,2],[6,3],[4,5]]

>>> reduce(lambda x,y: x+y,l)

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

16.Split the list into evenly sized chunks?

ex : if l=[1,2,3,4,5,6,7,8,9] o/p = [(1,2,3),(4,5,6),(7,8,9)]

import pprint

def chunks(l, n):

"""Yield successive n-sized chunks from l."""

for i in range(0, len(l), n):

yield tuple(l[i:i + n])

pprint.pprint(list(chunks(range(1, 10), 3)))

17.what is list comprehension and how does it works?

List comprehensions are a tool for transforming one list (any iterable actually) into another list. During this transformation,

elements can be conditionally included in the new list and each element can be transformed as needed.

If you’re familiar with functional programming, you can think of list comprehensions as syntactic sugar for a filter followed

by a map:

>>> doubled\_odds = map(lambda n: n \* 2, filter(lambda n: n % 2 == 1, numbers))

>>> doubled\_odds = [n \* 2 for n in numbers if n % 2 == 1]

18.How can you copy a dict into other dict?

dict1={"a":1,"b",2}

dict2=dict1

19.what is the output of the following code ?

s1="python"

s2=s1

s2.replace("p","j")

print s1,s2

o/P: pyhton python

20.what is the output of the following code ?

l=[1,2,3,4]

l1=l

l2=l[:]

l1.append(5)

print l

print l1

print l2

O/P : l =[1, 2, 3, 4, 5]

l1=[1, 2, 3, 4, 5]

l2=[1, 2, 3, 4]