Lists and Tuples in Python

BY ROHIT S. AGRAWAL

Lists in Python

- A list in Python is used to store the sequence of various types of data.
- Python lists are mutable type.
- A list can be defined as a collection of values or items of different types.
- The items in the list are separated with the comma (,) and enclosed with the square brackets [].
- List1 = [“Nashik”, ”Pune”, ”Mumbai”]
- List2 = [1, 2, 3, 4, 5, 6]
- List3 = [10,20,30,’Order’,’Product’,’Employee’]

Characteristics of Lists

- The lists are ordered.
- The element of the list can access by index.
- The lists are the mutable type.
- A list can store the number of various elements.

Lets try this

Try to manage the employee details along with the department and head of the department details using the list.

- Create a list Emp1 with employee details
- Dept list holds the department name and id
- Another list can contain the details of head of the department

List indexing

- The indexing is processed in the same way as it happens with the strings.
- The elements of the list can be accessed by using the slice operator [].
- The index starts from 0 and goes to length – 1
- mylist = [[‘Virat’ ,90],[‘Rohit’,85],30]
- print(mylist [0])
- print(mylist [0][0][3]+mylist[1][0][2])
- print(mylist [0][1] + mylist[1][1] / mylist[2])
- What is the output for the above print statements?

List Methods

- mylist.append(x)
- mylist.extend(c)
- mylist.pop()
- mylist.insert(i,x)
- mylist.remove(x)
- mylist.sort()
- mylist.reverse()
- mylist.index()
- mylist.count()

Updating List values

- Lists are the most versatile data structures in Python since they are mutable, and their values can be updated by using the slice and assignment operator.
- mylist = [1,12,32,45,50,60]
- try to assign a different value at index 3 in above list.
- How will you add multiple elements at a time to list?
- Try to add an element to the last of the list.
- Try to use + operator with the list to concatenate the list
- Try to use the (*) operator with the list (repetition)
- Try to add a List in the list

Removing List elements

- The list elements can also be deleted by using the **del** keyword.
- Python also provides us the **remove()** method if we do not know which element is to be deleted from the list.
- Delete one element from list – del mylist[2]
- Delete multiple elements from the list.
- Delete an element whose position is not available
- Delete the entire list

Removing List elements

- We can use remove() to remove the given item or pop() to remove an item at the given index.
- The pop() method removes and returns the last item if the index is not provided. This helps us implement lists as stacks.
- And, if we have to empty the whole list, we can use the clear() method.
- Delete element using pop() method
- Delete element using remove() method
- Delete element using pop() – pass argument as 1
- Clear the list using clear() method

List Slicing in Python

- Name = ‘Programming’   // given a string variable convert this to list
- Print elements from index 2 to index 5
- Print elements from index 5 to end
- Print last 3 elements(using negative index)
- Print elements from start to end
- Print alternate elements
- Print the list in reverse order using slice operator

Sorting

- Only Lists have built-in sorting method.
- Thus you often convert your data to the list if it needs sorting
- mylist = list(‘dbczbde’)
- print(mylist)
- mylist.sort()
- Convert this list to string again
- SortStr =‘ ‘.join(mylist)
- print(SortStr)

Guess the o/p

- mylist = [9,6,8,1,3,2]
- mylist = mylist.sort()
- print(mylist)

Built-in functions

- len(list)
- max(list)
- min(list)
- list(seq)
- Write the program to remove the duplicate element of the list.

Anagram Example

- Anagrams are words that contain the same letters in different order.
- For eg - cinema and iceman
- A strategy to identify anagrams is to take the letters of the word, sort those letters, then compare the sorted sequence
- Anagrams should have the same sequence
- Try solve one.

List Comprehension

- List comprehension is an elegant and concise way to create a new list from an existing list in Python.
- A list comprehension consists of an expression followed by for statement inside square brackets.
- pow2 = [2 ** x for x in range(10)]
- print(pow2)
- This code is equivalent to:
- pow2 =[]
- For x in range(10):
-     pow2.append(2**x)

Tuples in Python

- Python Tuple is used to store the sequence of immutable Python objects.
- A tuple can be written as the collection of comma-separated (,) values enclosed with the small () brackets.
- The parentheses are optional but it is good practice to use.
- A tuple can have any number of items and they may be of different types (integer, float, list, string, etc.).
- Create empty tuple
- Create tuple of numbers
- Create tuple of strings
- Create tuple of mixed types

Tuples…

- A tuple can also be created without using parentheses.
- This is known as tuple packing.

```
mytuple =3 , 8, “cat"
print(mytuple)
# tuple unpacking is also possible
a, b, c = mytuple
print(a)     # 3
print(b)     # 8
print(c)     # cat
```

Lets Create it..

- Try to create a tuple with single element.
- Check the type of the variable you created to confirm it’s a tuple.
- mytuple = (“Python”)
- print(mytuple)
- print(type(mytuple))

Indexing tuples

- We can use the index operator ([]) to access an item in a tuple, where the index starts from 0.
- So, a tuple having 6 elements will have indices from 0 to 5.
- The index must be an integer, so we cannot use float or other types.
- nested tuples are accessed using nested indexing.
- Try to print the 3rd element from the tuple
- Print 5th element from the tuple(index position does not exists)
- Try to access the index with float value
- Try to access a nested tuple elements
- Try to access the element using the negative index

Slicing tuple

- Name = ‘Programming’   // given a string variable convert this to tuple
- Print elements from index 2 to index 5
- Print elements from index 5 to end
- Print last 3 elements(using negative index)
- Print elements from start to end
- Print alternate elements
- Print the tuple in reverse order using slice operator

Changing a Tuple

- Unlike lists, tuples are immutable.
- This means that elements of a tuple cannot be changed once they have been assigned.
- Mytuple = (2,1,3,4,[6,5,7,8])
- Print(mytuple)
- Mytuple[2] = 23
- Mytuple[4][2] = 10
- Print(Mytuple)
- Try to reassign the tuple
- if the element is itself a mutable data type like a list, its nested items can be changed.(inside a tuple)

Operations with tuple

- Try to use the (+) operator with tuple
- Try to use the (*) repetition operator with tuple
- Try to delete element from a tuple
- Try to delete entire tuple
- Print the tuple just deleted
- Try to use the count() method with tuple
- Try to use the index() method with tuple

Iterating a tuple

- We can iterate through a tuple just like we do with list.
- We use looping constructs and conditional statements for the same.
- Membership operators are very handy while iterating a tuple
- Try to traverse through a tuple
- Check if the name exists in the tuple

Anagram Example

- Lets try our anagram example with the tuple