How to create tuple?

Tuples may be constructed in a number of ways:

1. Using a pair of parentheses to denote the empty tuple: ()
2. Using a trailing comma for a singleton tuple: a, or (a,)
3. Separating items with commas: a, b, c or (a, b, c)
4. Using the tuple() built-in: tuple() or tuple(iterable)

Example

```
>>> t1=()
>>> print(t1,type(t1))
() <class 'tuple'>
>>> t2=(10)
>>> print(t2,type(t2))
10 <class 'int'>
>>> t3=(10,)
>>> print(t3,type(t3))
(10,) <class 'tuple'>
>>> t4=(10,20,30,40,50)
>>> print(t4,type(t4))
(10, 20, 30, 40, 50) <class 'tuple'>
>>> t5=(10,1.5,1+2j,"naresh")
>>> print(t5,type(t5))
(10, 1.5, (1+2j), 'naresh') <class 'tuple'>
>>> t6=10,20,30,40,50
>>> print(t6,type(t6))
(10, 20, 30, 40, 50) <class 'tuple'>
>>> t7=10,
>>> print(t7,type(t7))
(10,) <class 'tuple'>
>>> t8=tuple()
>>> print(t8,type(t8))
() <class 'tuple'>
>>> t9=tuple(range(1,6))
>>> print(t9,type(t9))
(1, 2, 3, 4, 5) <class 'tuple'>
>>> t10=tuple("PYTHON")
>>> print(t10)
('P', 'Y', 'T', 'H', 'O', 'N')
>>> t11=tuple([10,20,30,40,50])
```

```
>>> print(t11,type(t11))
(10, 20, 30, 40, 50) <class 'tuple'>
```

What is difference between list and tuple?

List	Tuple
List is mutable collection	Tuple is immutable collection
After creating list changes can be done	After creating tuple changes cannot be done
List is created using []	Tuple is created using ()
List occupy more space	Tuple occupy less space
List cannot used as contained for other collections like set and dictionary	Tuple can be used as container for set and dictionary
List is not hashable object	Tuple is hashable object
“list” class represents list object	“tuple” class represents tuple object
Comprehension allowed	Comprehension not allowed

Reading Content of tuple

```
t1=(1,2,3,4,5,6,7,8,9,10)

print("Reading using Index ")
for i in range(len(t1)):
    print(t1[i],end=' ')

print()
print("Using Slicing")
t2=t1[:5]
print(t2)

print("Using for loop")
for value in t1:
    print(value,end=' ')

print()
print("Using Iterator")
a=iter(t1)
for value in a:
    print(value,end=' ')
```

```
print()
print("Using enumerate")
b=enumerate(t1)
for t in b:
    print(t)
```

Output

```
Reading using Index
1 2 3 4 5 6 7 8 9 10
Using Slicing
(1, 2, 3, 4, 5)
Using for loop
1 2 3 4 5 6 7 8 9 10
Using Iterator
1 2 3 4 5 6 7 8 9 10
Using enumerate
(0, 1)
(1, 2)
(2, 3)
(3, 4)
(4, 5)
(5, 6)
(6, 7)
(7, 8)
(8, 9)
(9, 10)
```

Example:

```
>>> x,y,z=A
>>> print(x,y,z)
10 20 30
>>> x,*y=A
>>> print(x,y)
10 [20, 30]
```

How to find size of tuple?

```
>>> t1=(10,20,30,40)
>>> import sys
>>> sys.getsizeof(t1)
```

Note: any object size is find in python using getsizeof() function available in sys module.

Example:

'''

Python – Maximum and Minimum K elements in Tuple
Sometimes, while dealing with tuples, we can have problem in which we need to extract only extreme K elements, i.e maximum and minimum K elements in Tuple.
This problem can have applications across domains such as web development and Data Science.

Input : test_tup = (3, 7, 1, 18, 9), k = 2
Output : (3, 1, 9, 18)

'''

```
test_tup=(3, 7, 1, 18, 9)
k=2
test_tup1=tuple(sorted(test_tup))
test_tup2=test_tup1[:k]+test_tup1[-k:]

print(test_tup)
print(test_tup1)
print(test_tup2)
```

'''

Python program to create a list of tuples from given list having number and its cube in each tuple

Given a list of numbers of list, write a Python program to create a list of tuples having first element as the number and second element as the cube of the number.

Example:

Input: list = [1, 2, 3]
Output: [(1, 1), (2, 8), (3, 27)]

'''

```
list1=[1,2,3]
list2=[(value,value**3) for value in list1]
print(list1)
print(list2)
```

Example:

```
# Python – Adding Tuple to List and vice – versa

list1=[10,20,30,40]
tuple1=(1,2,3,4,5)

list2=list1+list(tuple1)
tuple2=tuple1+tuple(list1)

print(list2)
print(tuple2)
```

'''

Python – Join Tuples if similar initial element

Input : test_list = [(5, 6), (5, 7), (5, 8), (6, 10), (7, 13)]
Output : [(5, 6, 7, 8), (6, 10), (7, 13)]

'''

```
t= [(5, 6), (5, 7), (5, 8), (6, 10), (7, 13)]
test_list1=[]

i=0
l=len(t)

while i<l:
```

```
j=i+1
while j<l:
    if t[i][0]==t[j][0]:
        x=tuple(set(t[i]+t[j]))
        del t[j]
        l=l-1
        continue
    j=j+1
```