```
# Finding size of tuple in python
# The size of a Tuple means the amount of memory (in bytes) taken by a Tuple object.

import sys

t1=(10,20,30,40,50)
b=sys.getsizeof(t1)
print(f'The size of tuple in bytes {b}')
t2=tuple(range(10,110,10))
b=sys.getsizeof(t2)
print(f'The size of tuple in bytes {b}')
print(t1,t2,sep="\n")
```

Output:

```
The size of tuple in bytes 80
The size of tuple in bytes 120
(10, 20, 30, 40, 50)
(10, 20, 30, 40, 50, 60, 70, 80, 90, 100)
```

sys.getsizeof(object)

getsizeof function of sys module return size of object or memory allocated for object in bytes.

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)

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

```
test_tup = (3, 7, 1, 18, 9)
test1=tuple(sorted(test_tup))

print(test_tup)
print(test1)
k=2
min_ele=test1[:k]
max_ele=test1[-k:]
test2=min_ele+max_ele
print(test2)
```

sorted() function

This function is used to sort element of iterable in ascending order or descending order. Sorted is immutable, after sorting iterale, it returns sorted elements in new iterable. This function is used for immutable collections.

```
>>> list1=[3,7,1,19,9]
>>> list1.sort()
>>> print(list1)
[1, 3, 7, 9, 19]
>>> t1=(3,7,1,19,9)
>>> t1.sort()
Traceback (most recent call last):
  File "<pyshell#4>", line 1, in <module>
    t1.sort()
AttributeError: 'tuple' object has no attribute 'sort'
>>> t2=sorted(t1)
```

```
>>> print(t2)
[1, 3, 7, 9, 19]
```

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)]

Input: list = [9, 5, 6]
Output: [(9, 729), (5, 125), (6, 216)]

```
list1=[1,2,3,4,5]
list2=[(num,num**3) for num in list1]
print(list1)
print(list2)
```

Python – Adding Tuple to List and vice – versa

Sometimes, while working with Python containers, we can have a problem in which we need to perform addition of one container with another. This kind of problem can have occurrence in many data domains across Computer Science and Programming.

```
list1=[10,20,30,40,50]
tuple1=(1,2,3,4,5)

list2=list1+list(tuple1)
print(list1)
```

```
print(tuple1)
print(list2)
tuple2=tuple(list1)
print(tuple2)
```

Output:

```
[10, 20, 30, 40, 50]
(1, 2, 3, 4, 5)
[10, 20, 30, 40, 50, 1, 2, 3, 4, 5]
(1, 2, 3, 4, 5, 10, 20, 30, 40, 50)
```

Python – Closest Pair to Kth index element in Tuple

Sometimes, while working with Python records, we can have a problem in which we need to find the tuple nearest to certain tuple, query on a particular index. This kind of problem can have applications in data domains such as web development.

Input : test_list = [(3, 4), (78, 76), (2, 3), (9, 8), (19, 23)] tup = (17, 23) K = 2
Output : (19, 23)
Input : test_list = [(3, 4, 9), (5, 6, 7)] tup = (1, 2, 5) K = 3
Output : (5, 6, 7)

What is difference between list and tuple?

List	Tuple
List is a mutable collection, after creating list changes can be done.	Tuple is immutable collection, after creating tuple changes cannot done.
List is created using []	Tuple is creating using ()

“list” type or class used for representing list object	“tuple” type of class used for representing tuple object
List occupy more space, because for performing mutable operations it required extra space	Tuple occupy less space, because it is immutable
In application development list is used to represent group of individual object where changes can be done.	In application development tuple is used to represent group of individual objects where changes cannot be done.
List is not hashable, it cannot used to represent data of set and dictionary.	Tuple is hashable, it can be used represent data of set and dictionary

String or str data type

String is collection of characters, these characters can be alphabets, digits or special characters.
String is an immutable sequence. After creating string object changes cannot be done.
String is non numeric data type, we cannot perform arithmetic operations.

How to create string?

1. String can be represented within single quotes ‘ ‘
2. String can be represented within double quotes” “

3. String can be represented within triple single or double quotes
   ‘’’ ‘’’ , “””” “”””
4. str() □ empty string
5. str(object) □ return string representation of input object

Example:

```
s1='Python'
s2="Python"
s3='''Python'''
s4="""Python"""
print(s1,s2,s3,s4,sep="\n")
print(type(s1),type(s2),type(s3),type(s4),sep="\n")

s5='python is "easy" programming language'
s6="python is 'easy' programming langauge"
s7='''python
is easy
programming
language'''
s8="""Python
is easy
programming
language"""
print(s5,s6,s7,s8,sep="\n")

s8=str()
print(s8,type(s8))
s9=str(45) # int -- string
s10=str(1.5) # float -- string
s11=str(1+2j) # complex -- string
s12=str(True) # Boolean --string
```

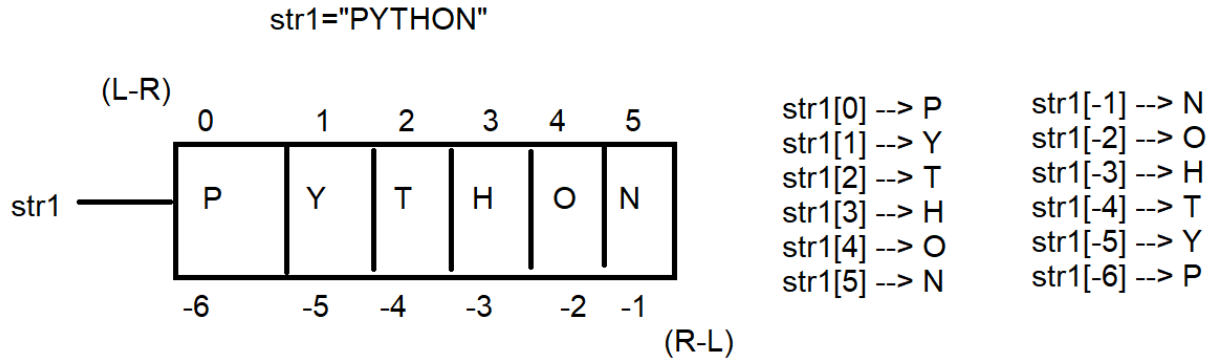
```
s13=str([10,20,30,40,50]) # list -- string
s14=str((1,2,3,4,5))
print(s9,s10,s11,s12,s13,s14,sep="\n")
print(type(s9),type(s10),type(s11),type(s12),type(s13),type(s14))

s15="PYTHON" # Alphabetic String
s16="PYTHON3" # Alphanumeric String
s17="1234" # Alphanumeric
print(s15,s16,s17,sep="\n")
```

Output:

```
Python
Python
Python
Python
<class 'str'>
<class 'str'>
<class 'str'>
<class 'str'>
python is "easy" programming language
python is 'easy' programming langauge
python
is easy
programming
language
Python
is easy
programming
language
 <class 'str'>
45
1.5
```

(1+2j)
True
[10, 20, 30, 40, 50]
(1, 2, 3, 4, 5)
<class 'str'> <class 'str'> <class 'str'> <class 'str'> <class 'str'> <class 'str'>
PYTHON
PYTHON3
1234



content of string can be read in different ways
1. index
2. slicing
3. iterator
4. enumerate
5. for loop

Example of immutable

```
>>> str1="PYTHON"
>>> del str1[0]
Traceback (most recent call last):
  File "<pyshell#8>", line 1, in <module>
    del str1[0]
TypeError: 'str' object doesn't support item deletion
>>> str1.append("3")
Traceback (most recent call last):
```

```
  File "<pyshell#9>", line 1, in <module>
    str1.append("3")
AttributeError: 'str' object has no attribute 'append'
>>> str1[0]="J"
Traceback (most recent call last):
  File "<pyshell#10>", line 1, in <module>
    str1[0]="J"
TypeError: 'str' object does not support item assignment
```

Example:

```
# write a program to count of character in input string

# str1="abc"
# c=3
# str2="abc123"
# c=6

str1=input("Enter any string ")
c=0
for s in str1:
    c=c+1

print(f'String is {str1}')
print(f'Count of characters {c}')
```

Output:

```
Enter any string abc123
String is abc123
Count of characters 6
```