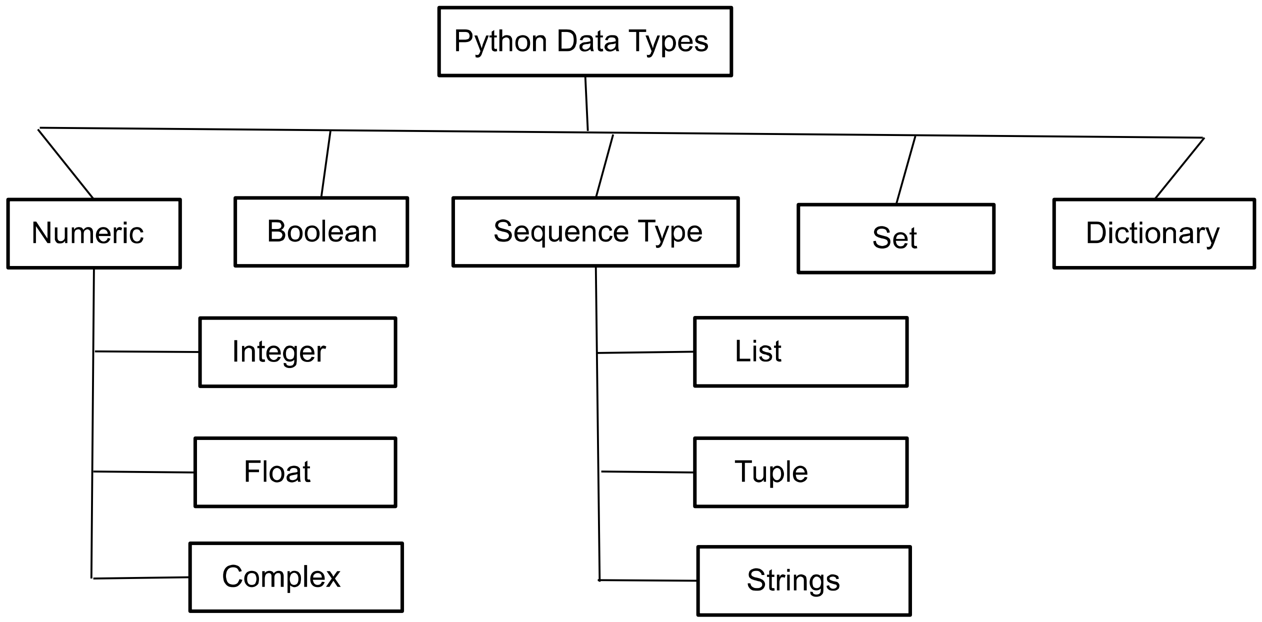
COMPONENTS OF PYTHON

Built in Object Types:

- All programming languages support a number of built-in variable types.These are the building blocks used for all the other variable types available within the language.
- By combining or extending the basic variable,we can create some complex variables.
- Python variables are actually objects.by using objects at the basic storage level the overall language is object based.This has a number of advantages.You can use the same methods and functions to operate on different types of objects.
- Python supports a wide variety of built-in object types that helps to make your programs easier to write.

Python supports five main built in object types



Numeric:

Numeric data type having the data which has numeric value i.e the numeric value can be integer,float,complex number.

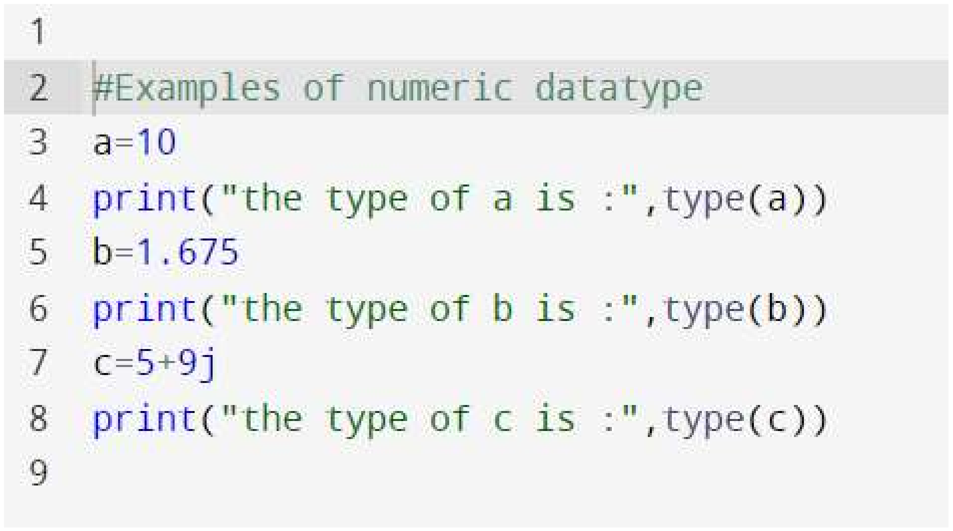
- Integers belongs to the class int
- Float belongs to the class float
- Complex belongs to the class complex

Integer:
It is represented by class int.It has the values as both positive and negative whole numbers.
Python has no restriction on how long an integer value can take.
Ex:10,20,99,-23,0,-100 etc.

Float:
It is represented by class float.it has the values with decimal point
It is accurate up to 15 decimal points.
Internally python stores floating- point values as C doubles,giving as much precision as possible.
Ex: 3.05,1.674,3.17

Complex:
It is represented by the class complex.
It is having two parts one is real and the other is imaginary
The form of complex number is x+iy.here x is real part and y is imaginary part
Ex: 2+5j, 3.0+5j

Python has a function named type function which is used to know the data type of any variable.

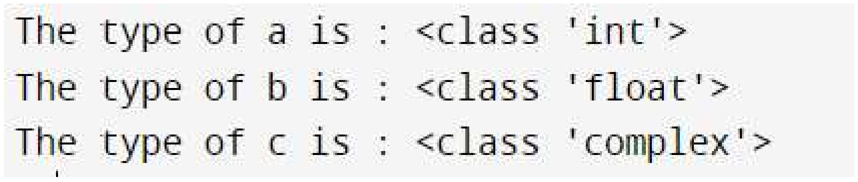


```python

#Examples of numeric datatype
a=10
print("the type of a is :",type(a))
b=1.675
print("the type of b is :",type(b))
c=5+9j
print("the type of c is :",type(c))

```

Output:



```
The type of a is : <class 'int'>
The type of b is : <class 'float'>
The type of c is : <class 'complex'>
```

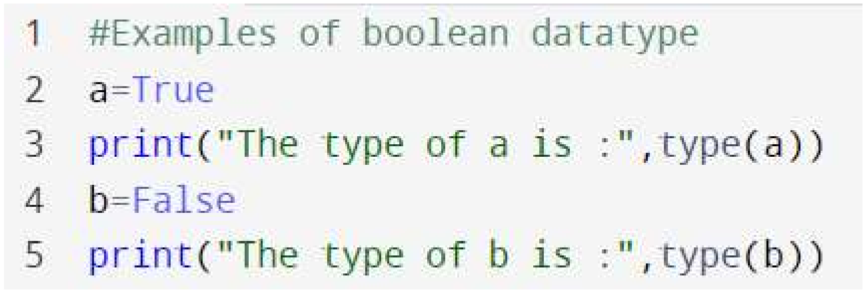
BOOLEAN:

It is represented by the class bool.
Boolean datatype provides only two values

1.True
2.False

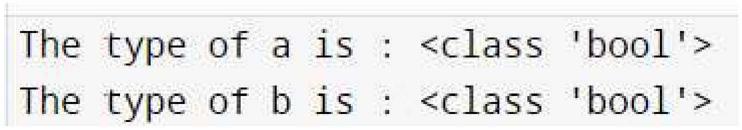
The first letter in both True and False must be capitalized otherwise the python interpreter will throw an error.
These values are used to determine the given statement true or false.

- True can be represented by any non-zero value or the letter 'T'
- False can be represented by the 0 or the letter 'F'.



```python
#Examples of boolean datatype
a=True
print("The type of a is :",type(a))
b=False
print("The type of b is :",type(b))
```

Output:



```
The type of a is : <class 'bool'>
The type of b is : <class 'bool'>
```

SET DATATYPE:

- It belongs to the class Set
- Python Set is the unordered collection of the data i.e it does not support indexing.
- sets can contain elements of Datatype int, string, Tuple, etc but can not contain List, sets, dictionaries.
- It is iterable and has only unique elements.Set data structure is used when we want each element to be unique in a large collection of data.

Properties of set datatype:

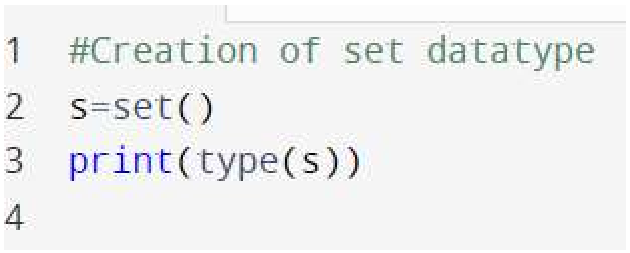
1.unordered
2.immutable
3.unique

1.unordered:
Set will store the element in an unordered manner such that no direct access to the element is possible. We use an iterator to iterate over all elements of the set.
2.immutable:
set elements are immutable i.e can't be changed but the set is mutable which means we can insert and delete elements from the set.
3.unique:
The frequency of each element in the set is always one.that means no duplicates are allowed to add to a set.

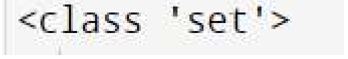
INITIALIZATION OF SET :

Set can be initialized in two ways
1.using set function
2.using curly braces { }

Using set function:
Initialization : using the inbuilt set function
Syntax: variable= set(iterable element)
Ex: s=set( )



```
#Creation of set datatype
s=set()
print(type(s))
```



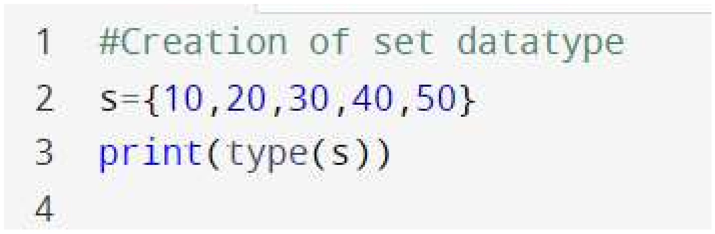
```
<class 'set'>
```

INPUT OUTPUT

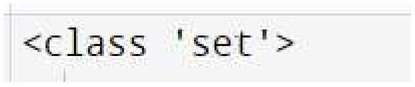
Using curly braces:

Syntax: variable={element1,element2,........element n}

Ex: s={10,20,30,40,50}



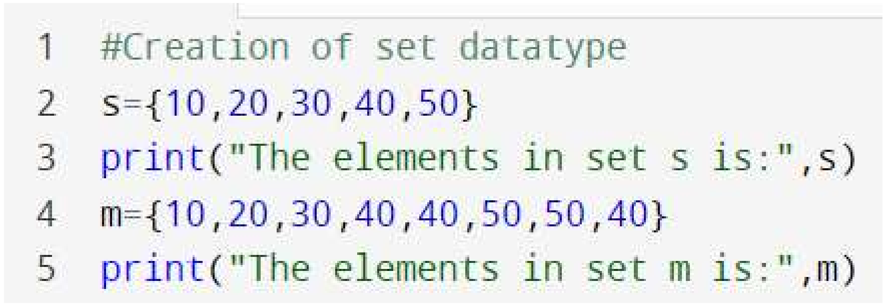
```python
#Creation of set datatype
s={10,20,30,40,50}
print(type(s))
```



```
<class 'set'>
```

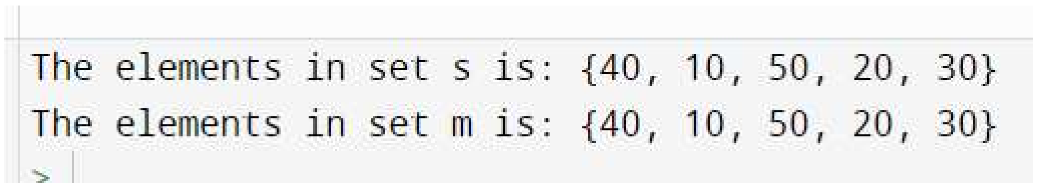
INPUT OUTPUT

While using this type of initialisation even if you give the duplicate elements it will not be taken into the set elements.
Ex:



```python
#Creation of set datatype
s={10,20,30,40,50}
print("The elements in set s is:",s)
m={10,20,30,40,40,50,50,40}
print("The elements in set m is:",m)
```

OUTPUT:



```
The elements in set s is: {40, 10, 50, 20, 30}
The elements in set m is: {40, 10, 50, 20, 30}
>
```

Some important points:
1.observe the elements order in set s and set m in both the input and output as we can clearly see that there is no order i.e unordered one of the properties of set .
2. See set m we have given the duplicate values but after the printing we get only unique elements in the set i.e unique elements one of the properties of set

DICTIONARY

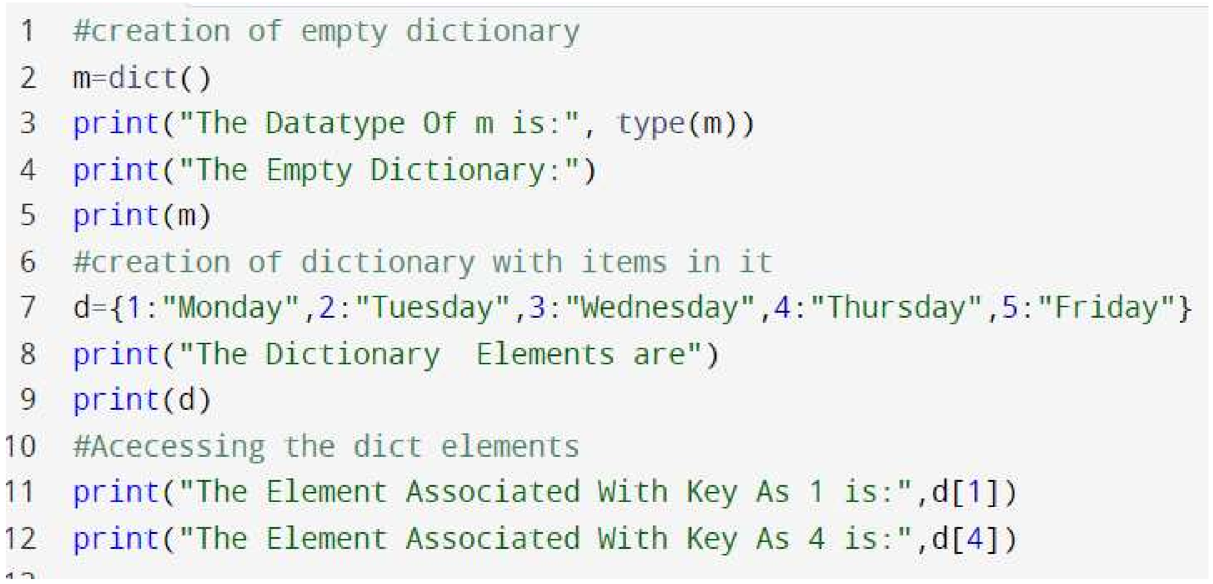
- It belongs to the class ‘dict’.
- Dictionary is an unordered set of a key-value pair of itemsDictionary can be created by placing a sequence of elements within curly {} braces, separated by comma in between the elements of the dictionary.
- It contains elements as key value pairs. Where keys can’t be repeated i.e unique and must be immutable i.e can't change.the values in a dictionary of any datatype and can be duplicated.
- The elements in the dictionary are accessed by using the key values,so the key values must be unique.
- Dictionary keys are case sensitive, i.e same name but different cases of Key will be treated differently.

Creation of a dictionary:

1.by using dict function.

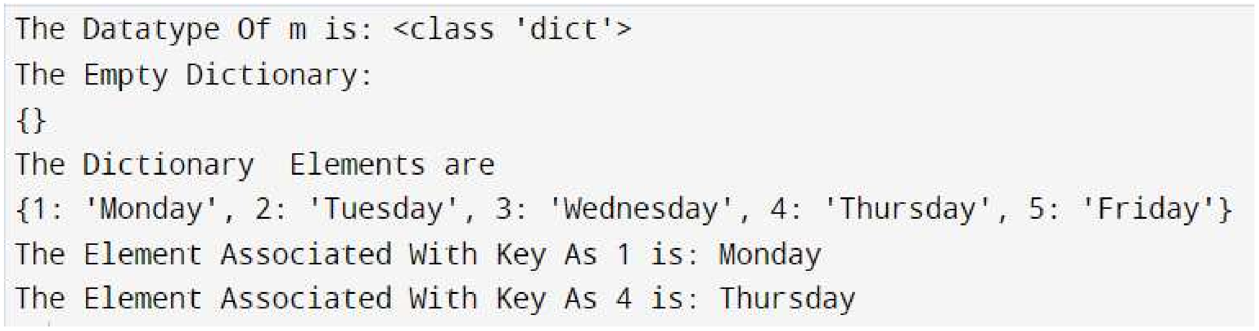
2.using curly braces which are separated by commas.

Ex:



```
#creation of empty dictionary
m=dict()
print("The Datatype Of m is:", type(m))
print("The Empty Dictionary:")
print(m)
#creation of dictionary with items in it
d={1:"Monday",2:"Tuesday",3:"Wednesday",4:"Thursday",5:"Friday"}
print("The Dictionary  Elements are")
print(d)
#Acecessing the dict elements
print("The Element Associated With Key As 1 is:",d[1])
print("The Element Associated With Key As 4 is:",d[4])
```

OUTPUT:



```
The Datatype Of m is: <class 'dict'>
The Empty Dictionary:
{}
The Dictionary  Elements are
{1: 'Monday', 2: 'Tuesday', 3: 'Wednesday', 4: 'Thursday', 5: 'Friday'}
The Element Associated With Key As 1 is: Monday
The Element Associated With Key As 4 is: Thursday
```

LIST DATATYPE

LIST : It belongs to class ‘list’

Python lists are similar to arrays in c language.But the other advantage of list is the elements of lists can be of any datatype.

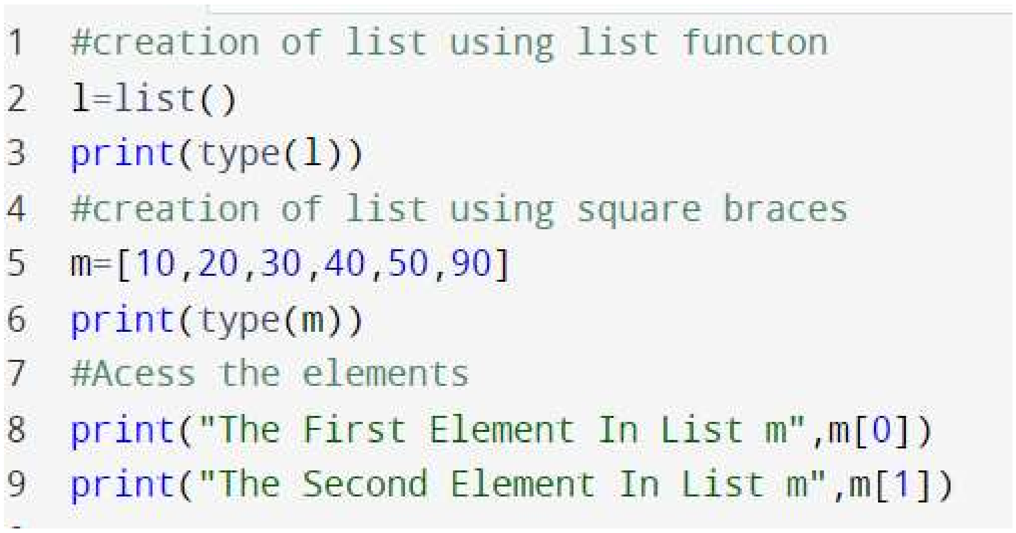
The items stored in the list are separated with a comma (,) and enclosed within square brackets [ ].

Creating a list in python:

1.by using list function

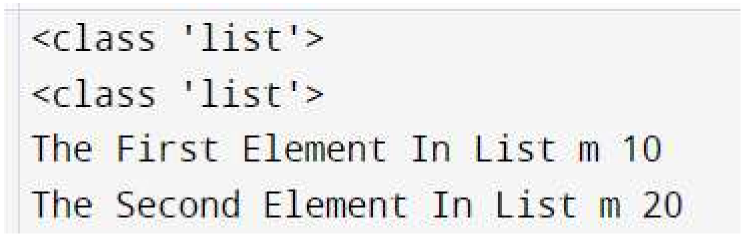
2.by using square braces [ ]

Ex:



```
#creation of list using list functon
l=list()
print(type(l))
#creation of list using square braces
m=[10,20,30,40,50,90]
print(type(m))
#Acess the elements
print("The First Element In List m",m[0])
print("The Second Element In List m",m[1])
```

OUTPUT:



```
<class 'list'>
<class 'list'>
The First Element In List m 10
The Second Element In List m 20
```

Access of elements in list:

- In order to access the list items refer to the index number. Use the index operator [ ] to access an item in a list.
- The elements in accessed by using the index.It starts with zero and end with (number of elements in list -1)
  Ex: l=[10,20,30,40,50,60,70,80,90]
- Number of elements in list l is 9
- Start index is 0 and end index is (9-1) i.e 8
- To access first element we use zero for last element 8

TUPLE DATATYPE

TUPLE: It belongs to class ‘tuple’

Tuple is an ordered collection of Python objects.

A tuple is similar to the list in . Like lists, tuples also contain the collection of the items of different data types. The items of the tuple are separated with a comma (,) and enclosed in parentheses ().

The only difference between tuple and list is that tuples are immutable i.e. tuples cannot be modified after it is created.

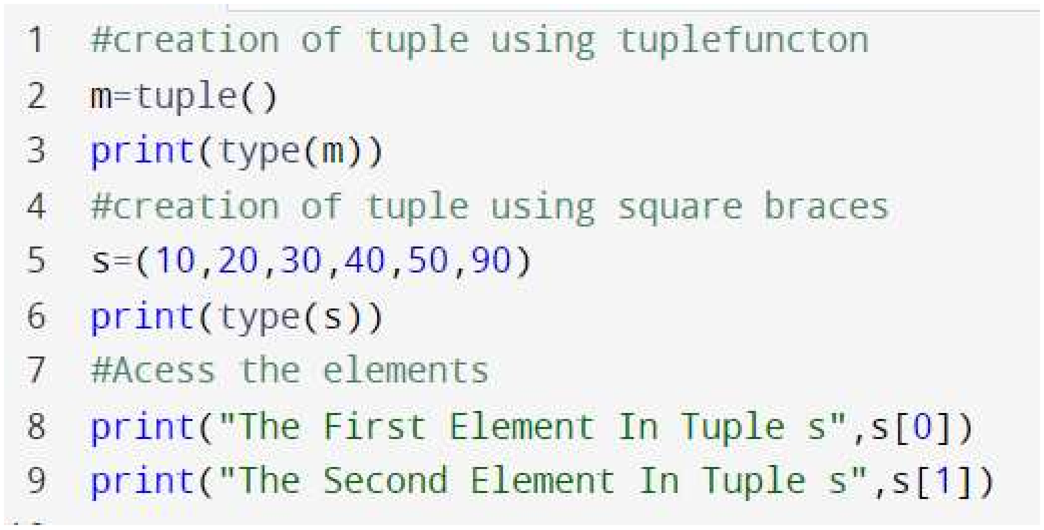
A tuple is a read-only data structure as we can't modify the value of tuple elements and size of a tuple.

Creating a tuple in python:

1.by using tuple function

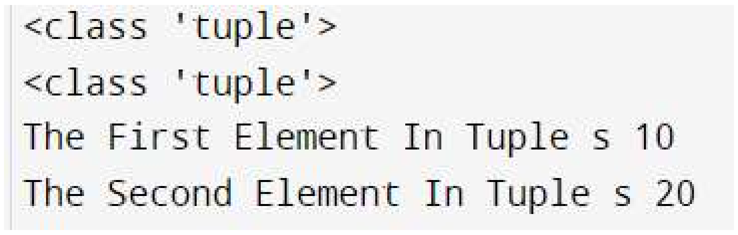
2.by using parentheses ().

Ex:



```
#creation of tuple using tuplefuncton
m=tuple()
print(type(m))
#creation of tuple using square braces
s=(10,20,30,40,50,90)
print(type(s))
#Acess the elements
print("The First Element In Tuple s",s[0])
print("The Second Element In Tuple s",s[1])
```

OUTPUT:

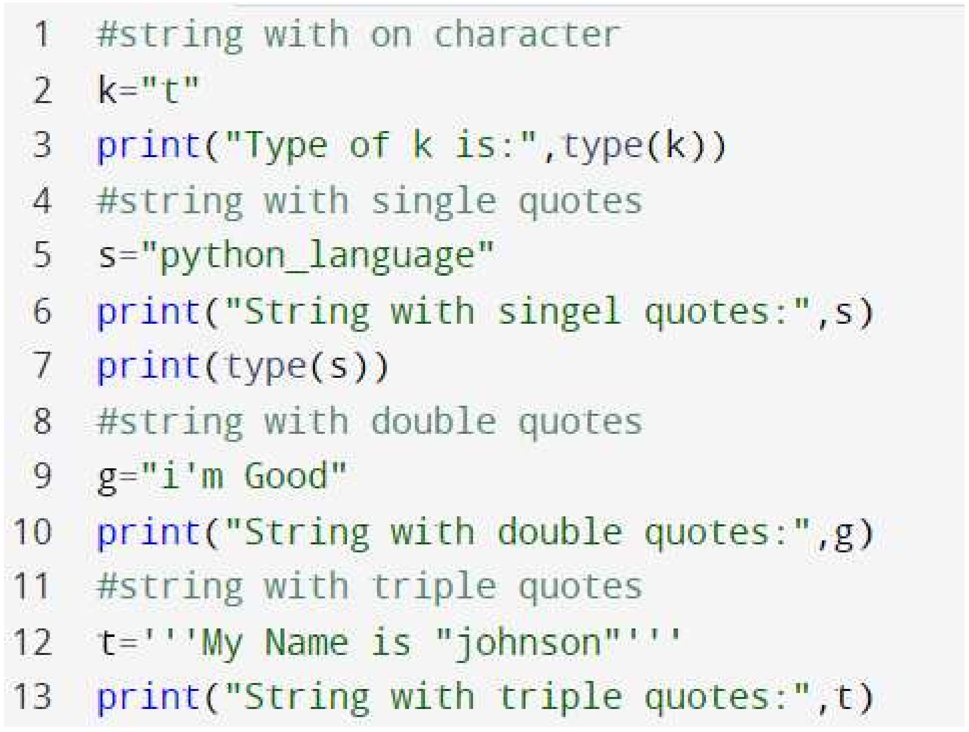


```
<class 'tuple'>
<class 'tuple'>
The First Element In Tuple s 10
The Second Element In Tuple s 20
```

Elements in tuple cant be modified

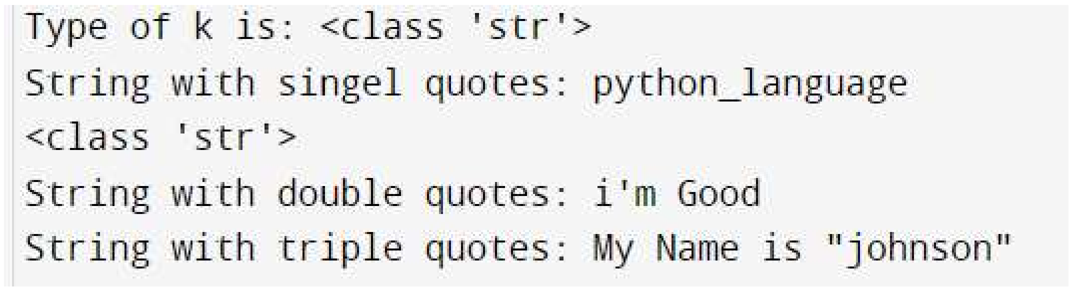
STRING DATATYPE

STRING: It belongs to class ‘str’.
string is a collection of one or more characters put in a single quote, double-quote or triple quote.
String handling in Python is a straightforward task since Python provides built-in functions and operators to perform operations in the string.
In python there is no character data type, a character is a string of length one.
When we write any thing in between single or double quotes or triple quotes it is string data type
Ex: s=”Python_is_user-Friendly”
Here s is a string datatype



```python
#string with on character
k="t"
print("Type of k is:",type(k))
#string with single quotes
s="python_language"
print("String with singel quotes:",s)
print(type(s))
#string with double quotes
g="i'm Good"
print("String with double quotes:",g)
#string with triple quotes
t='''My Name is "johnson"'''
print("String with triple quotes:",t)
```

OUTPUT:



```
Type of k is: <class 'str'>
String with singel quotes: python_language
<class 'str'>
String with double quotes: i'm Good
String with triple quotes: My Name is "johnson"
```