In [3]:
```
tup1 = ("Dev","Test","Prod","HR")
print(tup1)
print(tup1[3])
print(tup1[1:4])
for x in tup1:
    print(x)
for y in tup1:
    if(y == "Test"):
     print("Test is present")
```

```
('Dev', 'Test', 'Prod', 'HR')
HR
('Test', 'Prod', 'HR')
Dev
Test
Prod
HR
Test is present
```

In [12]:
```
tup2=(15,10,25,16,21,22,11,9,8,12,26,23,19)
print(tup2[1])
print(tup2[6])
for n in tup2:
    if(n==11):
        print(n, " is present in our tuple")
        print(str(n)+" is present in our tuple")
    else:
        print("*********")
        print(n)
```

```
10
11
*********
15
*********
10
*********
25
*********
16
*********
21
*********
22
11  is present in our tuple
11 is present in our tuple
*********
9
*********
8
*********
12
*********
26
*********
23
*********
19
```

In [13]:
```
tup1 = ("Dev","Test","Prod","HR")
tup1[3]="Functions"
print(tup1)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[13], line 2
      1 tup1 = ("Dev","Test","Prod","HR")
----> 2 tup1[3]="Functions"
      3 print(tup1)

TypeError: 'tuple' object does not support item assignment
```

In [14]:
```
tup1 = ("Dev","Test","Prod","HR")
del tup1[2]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[14], line 2
      1 tup1 = ("Dev","Test","Prod","HR")
----> 2 del tup1[2]

TypeError: 'tuple' object doesn't support item deletion
```

In [15]:
```
tup1 = ("Dev","Test","Prod","HR")
del tup1
print(tup1)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[15], line 3
      1 tup1 = ("Dev","Test","Prod","HR")
      2 del tup1
----> 3 print(tup1)

NameError: name 'tup1' is not defined
```

In [19]:
```
#Set
setOfNumbers={2,3,1,3,11,9,4,5,5,4}
print(setOfNumbers)
print(type(setOfNumbers))
```

```
{1, 2, 3, 4, 5, 9, 11}
<class 'set'>
```

In [20]:
```
#Union of two sets
A={1,2,3,4}
B={3,4,5,6}
print(A|B)
```

```
{1, 2, 3, 4, 5, 6}
```

In [21]:
```
#Intersection
A={1,2,3,4}
B={3,4,5,6}
print(A&B)
```

```
{3, 4}
```

In [22]:
```
#Symmetric Difference
A={1,2,3,4}
```

```
B={3,4,5,6}
print(A^B)
```

```
{1, 2, 5, 6}
```

In [25]:
```
#Accessing set elements
X={11,12,13,0,1,2,3,4,5,6,7,8,9,10}
#Using for loop
for x in X:
    if(x == 0):
        print(x," is present")
#Using in operator
if 0 in X:
    print("0 is present using if condition ")
```

```
0  is present
0 is present using if condition
```

In [27]:
```
#Accessing set elements
X={11,12,13,0,1,2,3,4,5,6,7,8,9,10}
print(len(X))
print(X[3])#Using index you can't access set elements
```

```
14
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[27], line 4
      2 X={11,12,13,0,1,2,3,4,5,6,7,8,9,10}
      3 print(len(X))
----> 4 print(X[3])

TypeError: 'set' object is not subscriptable
```

In [31]:
```
setOfData={"Hello","Happy","Republic","Day",26,"January",10.50}
print(setOfData)
print(dir(setOfData))
```

```
{'Hello', 'Republic', 26, 'Happy', 'Day', 10.5, 'January'}
['__and__', '__class__', '__class_getitem__', '__contains__', '__delattr__', '__dir__', '__doc__', '__eq__', '__format__',
'__ge__', '__getattribute__', '__getstate__', '__gt__', '__hash__', '__iand__', '__init__', '__init_subclass__', '__ior_
_', '__isub__', '__iter__', '__ixor__', '__le__', '__len__', '__lt__', '__ne__', '__new__', '__or__', '__rand__', '__reduc
e__', '__reduce_ex__', '__repr__', '__ror__', '__rsub__', '__rxor__', '__setattr__', '__sizeof__', '__str__', '__sub__',
'__subclasshook__', '__xor__', 'add', 'clear', 'copy', 'difference', 'difference_update', 'discard', 'intersection', 'inte
rsection_update', 'isdisjoint', 'issubset', 'issuperset', 'pop', 'remove', 'symmetric_difference', 'symmetric_difference_u
pdate', 'union', 'update']
```

In [39]:
```
setOfData={"Hello","Happy","Republic","Day",26,"January",10.50}
print(setOfData)
setOfData.add("Ramesh")#Add element in a set
print(setOfData)
copiedData = setOfData.copy()
print(copiedData)
copiedData.remove("Ramesh") # Removes a specified elements from set
print(copiedData)
print("************Update**************")
copiedData.update("15")
print(copiedData)
newSet = {15,10,25}
copiedData.update(newSet)# update function adds data from another set to a set
print(copiedData)
listOfNumbers = [21,22,24]
copiedData.update(listOfNumbers)#adds data from another list to a set
print(copiedData)
```

```
{'Hello', 'Republic', 26, 'Happy', 'Day', 10.5, 'January'}
{'Hello', 'Republic', 26, 'Ramesh', 'Happy', 'Day', 10.5, 'January'}
{'Hello', 10.5, 'Republic', 'Ramesh', 'Happy', 'Day', 26, 'January'}
{'Hello', 10.5, 'Republic', 'Happy', 'Day', 26, 'January'}
************************
{'Hello', '5', 10.5, 'Republic', 'Happy', 'Day', 26, 'January', '1'}
{'Hello', '5', 10.5, 10, 15, 'Republic', 'Happy', 'Day', 26, 'January', 25, '1'}
{'Hello', '5', 10.5, 10, 15, 'Republic', 21, 22, 24, 'Happy', 'Day', 26, 'January', 25, '1'}
```

In [49]:
```
#Another way to create a set
set1=set({5,6,7,4,2,1})
print(set1)
#Remove a element
set1.remove(6)
print(set1)
set1.discard(4)
print(set1)
#del set1[2] # 'set' object doesn't support item deletion
print("***************")
set1.pop() # Removes random element or items from a set
print(set1)
x=set1.pop()# Removes random element or items from a set and returns or gives you the element
print(x)
print(set1)
```

```
{1, 2, 4, 5, 6, 7}
{1, 2, 4, 5, 7}
{1, 2, 5, 7}
***************
{2, 5, 7}
2
{5, 7}
```

In [59]:

```python
setOfBooks=set({"Python Programming","Strategy","WSJ","Philip Kotler","OB"})
print(setOfBooks)
#pop
popedBook= setOfBooks.pop()
print(popedBook)
print("*************Before remove*********")
print(setOfBooks)
#remove
removedBook = setOfBooks.remove("Strategy")# Remove function removes random item and returns None if item is present else
print(removedBook)
print(setOfBooks)
print("*********Discard***************")
#discard
discardedBook=setOfBooks.discard("Java Programming")
print(discardedBook)
print(setOfBooks)
print("*******************Delete************")
del setOfBooks #deletes the entire set object itself
print(setOfBooks)
```

```
{'Philip Kotler', 'OB', 'Strategy', 'WSJ', 'Python Programming'}
Philip Kotler
*************Before remove*********
{'OB', 'Strategy', 'WSJ', 'Python Programming'}
None
{'OB', 'WSJ', 'Python Programming'}
*********Discard***************
None
{'OB', 'WSJ', 'Python Programming'}
*******************Delete************
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[59], line 19
     17 print("********************Delete************")
     18 del setOfBooks
---> 19 print(setOfBooks)

NameError: name 'setOfBooks' is not defined
```

In [60]:
```
setOfX ={1,2,3,4,5}
print(setOfX)
setOfX.clear() #Removes all items or elements from set.
print(setOfX)
```

```
{1, 2, 3, 4, 5}
set()
```

In [61]:
```
set1 = {'a','b','c','d','e','a'}
set2 = {'d','e','f','g','h'}
set3 = {'c','d','e'}
print(set1.union(set2))
print(set1.intersection(set2))
print(set1.issuperset(set3))
print(set3.issubset(set1))
print(set2.issuperset(set3))
set1.update(set2)
print(set1)
```

```
{'d', 'h', 'c', 'g', 'a', 'e', 'f', 'b'}
{'d', 'e'}
True
True
False
{'d', 'h', 'c', 'g', 'a', 'e', 'f', 'b'}
```

In [65]:
```
listOfNumbers =[4,5,6,7,8,9,10,1,2,3]
lengthOflistOfNumbers= len(listOfNumbers)
for index,n in enumerate(listOfNumbers):
  print("Index position "+str(index)+" contains value "+ str(n))
```

```
Index position 0 contains value 4
Index position 1 contains value 5
Index position 2 contains value 6
Index position 3 contains value 7
Index position 4 contains value 8
Index position 5 contains value 9
Index position 6 contains value 10
Index position 7 contains value 1
Index position 8 contains value 2
Index position 9 contains value 3
```

In [66]:
```
num=12
for i in range(1,11):
    print(num,'*',i,'=',num*i)
```

```
12 * 1 = 12
12 * 2 = 24
12 * 3 = 36
12 * 4 = 48
12 * 5 = 60
12 * 6 = 72
12 * 7 = 84
12 * 8 = 96
12 * 9 = 108
12 * 10 = 120
```

In [67]:
```
print(range(1,11))
```

```
range(1, 11)
```

In [72]:
```
print("Enter the number ")
num=int(input())
#num1= input()#input function takes values from standard input console in the form string
#print(type(num1))
#num=input()
for i in range(1,11):
    print(num,'*',i,'=',num*i)
```

```
Enter the number
11
11 * 1 = 11
11 * 2 = 22
11 * 3 = 33
11 * 4 = 44
11 * 5 = 55
11 * 6 = 66
11 * 7 = 77
11 * 8 = 88
11 * 9 = 99
11 * 10 = 110
```

In [ ]: