Dictionaries

In [52]:
```
#here we get by key not by index, since dictionary do not have fixed index value
#for key there must be a value,otherwise the default is none
#mutable
#key and value pair is item
```

Out[52]:
```
['__class__',
 '__contains__',
 '__delattr__',
 '__delitem__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__setattr__',
 '__setitem__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
 'clear',
 'copy',
 'fromkeys',
 'get',
 'items',
 'keys',
 'pop',
 'popitem',
 'setdefault',
 'update',
 'values']
```

In [53]:
```
d={"name":"p.sindhu","roll":"3","age":19}
d.values()
d.keys()
```

Out[53]:
```
dict_items([('name', 'p.sindhu'), ('roll', '3'), ('age', 19)])
```

In [54]:
```
```

Out[54]:
```
'p.sindhu'
```

In [55]:
```
```

Out[55]:
```
{'name': 'divija', 'roll': '3', 'age': 19}
```

In [57]:
```
d["gender"]="F"
```

Out[57]:
```
{'name': 'divija', 'roll': '3', 'age': 19, 'gender': 'F'}
```

In [64]:
```
#using get method
d.get("name")
```

Out[64]:
```
'g.divija'
```

In [59]:
```
d.update({"name":"g.divija"})
d
```

Out[59]:
```
{'name': 'g.divija', 'roll': '3', 'age': 19, 'gender': 'F'}
```

In [60]:
```
d.setdefault("v")
d
#none in output represents that it donot contains any value
```

Out[60]:
```
{'name': 'g.divija', 'roll': '3', 'age': 19, 'gender': 'F', 'v': None}
```

In [61]:
```
d["a"]="a"
d
```

Out[61]:
```
{'name': 'g.divija',
 'roll': '3',
 'age': 19,
 'gender': 'F',
 'v': None,
 'a': 'a'}
```

In [66]:
```
d.setdefault("p")
d
```

Out[66]:
```
{'name': 'g.divija',
 'roll': '3',
 'age': 19,
 'gender': 'F',
 'v': None,
 'a': 'a',
 'p': None}
```

In [136]:
```
d["p"]="divija"
d
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-136-f761e7257b64> in <module>()
----> 1 d["p"]="divija"
      2 d

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

In [69]:
```
l=[1,2]
a=[5,5]
x=dict(zip(a,l))
x
```

Out[69]:
```
{5: 2}
```

In [43]:
```
#*

##check whether the given no. is in range or not
##Lb=Lower bound
##Ub=upper bound
##lb=20,ub=30
def check_range():
    if(n>lb and n<ub):
        print("a,b")
    else:
        print("a,b")
lb=int(input("lb"))
ub=int(input("ub"))
n=int(input("not in range"))
print(check_range())
```

```
lb8
ub94
not in range55
a,b
None
```

In [65]:
```
#calculate no. of digits
##i/p:12345 o/p:5
def count1(n):
    #n=int(input("Enter number:"))
    count=0
    while(n!=0):
        r=n%10
        count=count+1
        n=n//10
    print("The number of digits in the number are:",count)
count1(12345)
#or
len(str("we"))

```

```
The number of digits in the number are: 5
```

Out[65]:
```
2
```

In [69]:
```
# Print the leap years within the given range
#lowerbound=1990
#upperbound=2020
def leapyear(i):
    lb=int(input("enter lowerbound"))
    ub=int(input("enter upperbound"))
    for i in range(lb,ub):
        if(i%400==0)or(i%4==0 and i%100!=0):
            print(i)
leapyear(2020)

```

```
enter lowerbound1990
enter upperbound2020
1992
1996
2000
2004
2008
2012
2016
```

In [59]:
```
#**
# Python program to count the
# number of numbers in a given range
# using traversal and mutliple line code

def count(list1, l, r):
c=0
# traverse in the list1
 for x in list1:
    if x>= l and x<= r:
        c=c+1
         return(c)
    list1 = [10, 20, 30, 40, 50, 40, 40, 60, 70]
    l = 40
    r = 80
    print count(list1, l, r)
```

```
  File "<ipython-input-59-bd6f3be5b459>", line 6
    c=0
    ^
IndentationError: expected an indented block
```

In [141]:
```
#**
##check whether the given is in range or not
##Lb=lower bound
##Ub=upper bound
##lb=20,ub=30
def check_range():
    if(n>=lb and n<=ub):
        print(n,"n is in range")
    else:
        print(n,"n is not in range")
check_range(lb,ub)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-141-86b982c688a9> in <module>()
      9     else:
     10         print(n,"n is not in range")
---> 11 check_range(lb,ub)

TypeError: check_range() takes 0 positional arguments but 2 were given
```

In [73]:
```
#prime
def isprime(n):
    count=0
    for i in range(2,n//2+1):
        if(n%i==0):
            count+=1
    if count==0:
        return True
    else:
        return False
isprime(7)

```

Out[73]: True

In [82]:
```
##print the prime no.s in given range(2,50)
def prime(lb,ub):
    for i in range(lb,ub+1):
        if isprime(i):
            print(i)
prime(2,50)

```

```
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
```

In [84]:
```
#check if a given string equal to number
n="one"
s=1
if str(s)==n:#here we use type conversion,since we cannot compare int and strings
    print(True)
else:
    print(False)
```

```
False
```

In [91]:
```
##perfect numbers
#i/p:6 1,2,3   (1+2+3=6)
def perfect_number(n):
    sum = 0
    for x in range(1, n):
        if n % x == 0:
            sum += x
    return sum == n
print(perfect_number(496))
```

```
True
```

In [142]:
```
def perfect_number(n):
    sum = 0
    for x in range(1, 1000):
        if n % x == 0:
            sum += x
    return sum == n
print(perfect_number(6))
```

```
False
```

In [144]:
```
##**
def per(lb,ub):
    lb=int(input(i))
    ub=int(input(i))
    for i in range(lb,ub+1):
        if(per1(i)):
            print(i)
per(lb,ub)

```

```
---------------------------------------------------------------------------
UnboundLocalError                         Traceback (most recent call last)
<ipython-input-144-806d363e1d76> in <module>()
      5         if(per1(i)):
      6             print(i)
----> 7 per(lb,ub)

<ipython-input-144-806d363e1d76> in per(lb, ub)
      1 def per(lb,ub):
----> 2     lb=int(input(i))
      3     ub=int(input(i))
      4     for i in range(lb,ub+1):
      5         if(per1(i)):

UnboundLocalError: local variable 'i' referenced before assignment
```

In [119]:
```
#**
##given 2 var if one of them is 10 or their sum equals to Return True else return
def variables(a,b):
    if(a==10 or b==10):
        return True
    elif (a+b=10):
         return True
    else:
         return False
variables(5,5)
```

```
  File "<ipython-input-119-88559a68a766>", line 5
    elif (a+b=10):
               ^
SyntaxError: invalid syntax
```

In [120]:
```
#concatinating 2 strs
#using  "+"operand
#o/p: hello
```

In [132]:
```
#li=[10,9,8,7,6]
#sort list in asc order=[6,7,8,9,10]
#max ele in list
#min ele in list
##2nd largest no. in list
li=[10,9,8,7,6]
print(min(li))
```

```
6
```

In [127]:
```
li=[10,9,8,7,6]
max(li)
```

Out[127]:
```
10
```

In [129]:
```
li.sort()
li
```

Out[129]:
```
[6, 7, 8, 9, 10]
```

In [130]:
```

```

Out[130]:
```
9
```

In [135]:
```
#method2
def elements():
    num=[1,2,3,4,5]
    num.sort()
    print(num)
    print(max(num))
    print(min(num))
    print(num[-2])
elements()
```

```
[1, 2, 3, 4, 5]
5
1
4
```

In [ ]:
```

```