immutable data structures

- list is a mutable data structure it has following feature
- 1.the element can be chnage in place
- 2.element can be hetrogenious
- 3.element can be dublicate
- 4.it can be index,slice or loop throgh
- 5.the list can be nested

creating list

In [3]:
```
l=[1,2,3]
print(l)
print(type(l))
m=[]
print(type(m))
print(m)
n=list()
print(n)
print(type(n))
```

```
[1, 2, 3]
<class 'list'>
<class 'list'>
[]
[]
<class 'list'>
```

using eval

In [4]:
```
user=eval(input("enter list of a comma separated values-"))
print(user)
print(type(user))
```

```
enter list of a comma separated values-[1,2,3]
[1, 2, 3]
<class 'list'>
```

mutability

In [5]:
```
l=[1,2,3,4,5]
l[0]
```

Out[5]: 1

In [6]:
```
l[0]=100
print(l)
```

```
[100, 2, 3, 4, 5]
```

In [7]:
```
l=list(1,2,3,4,5)
print(l)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-7-17bd5a2b52f9> in <module>
----> 1 l=list(1,2,3,4,5)
      2 print(l)

TypeError: list expected at most 1 argument, got 5
```

In [8]:
```
l=list((1,2,3,4,5))
print(l)
```

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

In [9]:
```
s='python'
l=list(s)
print(l)
```

```
['p', 'y', 't', 'h', 'o', 'n']
```

In [10]:
```
s='python'
l=list(s)
print(l[0])
```

```
p
```

mixing diffrent data type

In [11]:
```
l=[1,2,None,(4,5,6),'hello',False]
print(l)
```

```
[1, 2, None, (4, 5, 6), 'hello', False]
```

indexing and slicing (same as tupple)

In [12]:
```
l[0]
```

Out[12]:
```
1
```

In [13]:
```
l[3]
```

Out[13]:
```
(4, 5, 6)
```

In [14]:
```
l[3][1]
```

Out[14]: 5

In [15]:
```
l[3][1]=100
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-15-c04068964e13> in <module>
----> 1 l[3][1]=100

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

In [16]:
```
l[4][2]
```

Out[16]: 'l'

In [17]:
```
l[2]=100
l
```

Out[17]: [1, 2, 100, (4, 5, 6), 'hello', False]

In [18]:
```
l[::-1] # reverse list
```

Out[18]: [False, 'hello', (4, 5, 6), 100, 2, 1]

In [19]:
```
l[:] # copy of list
```

Out[19]: [1, 2, 100, (4, 5, 6), 'hello', False]

In [20]:
```
l=[1,2,3,4,5,6,7]
l[1::2]
```

Out[20]: [2, 4, 6]

In [21]:
```
l[-2:-4]
```

Out[21]: []

In [22]:
```
l[-2:-4:-1]
```

Out[22]: [6, 5]

In [23]:
```
l[-4:-2]
```

Out[23]: [4, 5]

In [24]:
```
l[-4:-2:-1]
```

Out[24]: []

In [25]:
```
1 print(l)
```

```
[1, 2, 3, 4, 5, 6, 7]
```

In [26]:
```
1 l[4:2:-1] # l=[1,2,3,4]
```

Out[26]:
```
[5, 4]
```

List is slow And Dyanamic

In [27]:
```
1 l=[3,4,5,6] # list is dynamic
2 print(id(l[0]))
```

```
140726725846880
```

In [28]:
```
1 l=[3,4,5,6]
2 print(id(l))
```

```
1931344349696
```

In [29]:
```
1 print(id(l[1]))
```

```
140726725846912
```

In [30]:
```
1 print(id(l[2]))
```

```
140726725846944
```

list dimensions

In [31]:
```
1 l=[1,2,3]
2 m=[[1,2],[3,4]]
3 n=[[[1,2,3],[4,5,6]]]
4 print(l)
5 print(m)
6 print(n)
```

```
[1, 2, 3]
[[1, 2], [3, 4]]
[[[1, 2, 3], [4, 5, 6]]]
```

In [32]:
```
1 whos
```

```
Variable   Type    Data/Info
---------------------------
l          list    n=3
m          list    n=2
n          list    n=1
s          str     python
user       list    n=3
```

list method

append

In [33]:
```
l=[1,2,3,45]
l.append??
```

In [34]:
```
l.append(3)
l
```

Out[34]:
```
[1, 2, 3, 45, 3]
```

In [35]:
```
l.append((3,4))
l
```

Out[35]:
```
[1, 2, 3, 45, 3, (3, 4)]
```

In [36]:
```
l.append("python")
l
```

Out[36]:
```
[1, 2, 3, 45, 3, (3, 4), 'python']
```

In [37]:
```
l.append(6,7,8)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-37-685c9db41dd1> in <module>
----> 1 l.append(6,7,8)

TypeError: append() takes exactly one argument (3 given)
```

In [38]:
```
l.append((6,7,8))
l
```

Out[38]:
```
[1, 2, 3, 45, 3, (3, 4), 'python', (6, 7, 8)]
```

extend

In [39]:
```
l=[3,4,5,6]
```

In [40]:
```
l.extend??
```

In [41]:
```
l.extend('python')
```

In [42]:
```
l
```

Out[42]:
```
[3, 4, 5, 6, 'p', 'y', 't', 'h', 'o', 'n']
```

In [43]:
```
l.extend((1,2,3))
l
```

Out[43]: [3, 4, 5, 6, 'p', 'y', 't', 'h', 'o', 'n', 1, 2, 3]

In [44]:
```
l.extend(range(5))
```

In [45]:
```
l
```

Out[45]: [3, 4, 5, 6, 'p', 'y', 't', 'h', 'o', 'n', 1, 2, 3, 0, 1, 2, 3, 4]

insert

In [46]:
```
l=[4,5,6,7]
l.insert??

```

In [47]:
```
l.insert(2,100)
```

In [48]:
```
l
```

Out[48]: [4, 5, 100, 6, 7]

In [49]:
```
l.insert(1,[1,2])
l
```

Out[49]: [4, [1, 2], 5, 100, 6, 7]

editing the list

In [50]:
```
l
```

Out[50]: [4, [1, 2], 5, 100, 6, 7]

In [51]:
```
l[2:]=[12,13,14,15]
```

In [52]:
```
l
```

Out[52]: [4, [1, 2], 12, 13, 14, 15]

add and multiplying lists

In [53]:
```
l1=[1,2,3]
l2=list('python')
l3=[4,5,6]
l1+l2+l3
```

Out[53]: [1, 2, 3, 'p', 'y', 't', 'h', 'o', 'n', 4, 5, 6]

In [54]:
```
l1=[1,2,3]
l2=list('python')
l3=[4,5,6]
l1*l2*l3
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-54-e9669ebe026c> in <module>
      2 l2=list('python')
      3 l3=[4,5,6]
----> 4 l1*l2*l3

TypeError: can't multiply sequence by non-int of type 'list'
```

In [55]:
```
l1*3
```

Out[55]:
```
[1, 2, 3, 1, 2, 3, 1, 2, 3]
```

delete method

In [56]:
```
l=[4,5,6,7]
del l
print(l)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-56-d63cf28c1706> in <module>
      1 l=[4,5,6,7]
      2 del l
----> 3 print(l)

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

In [57]:
```
l=[4,5,6,7]
m=l
del m
print(l)
```

```
[4, 5, 6, 7]
```

In [58]:
```
l=[4,5,6,7]
m=l
print(l)
print(m)
print(id(m))
print(id(l))
```

```
[4, 5, 6, 7]
[4, 5, 6, 7]
1931345737408
1931345737408
```

In [59]:
```
l=[4,5,6,7]
m=l[:]
print(l)
print(m)
print(id(m))
print(id(l))
```

```
[4, 5, 6, 7]
[4, 5, 6, 7]
1931345739712
1931343742336
```

In [60]:
```
l=[4,5,6]
m=l[:]
del m
print(l)
```

```
[4, 5, 6]
```

In [61]:
```
del l[0]
l

```

Out[61]:
```
[5, 6]
```

In [62]:
```
whos
```

```
Variable   Type    Data/Info
---------------------------
l          list    n=2
l1         list    n=3
l2         list    n=6
l3         list    n=3
n          list    n=1
s          str     python
user       list    n=3
```

remove

In [63]:
```
l=[6,7,8,9]
l.remove??

```

In [64]:
```
l=[6,7,8,9,6]
```

In [65]:
```
l.remove(6)

```

In [66]:
```
print(l)
```

```
[7, 8, 9, 6]
```

pop method

In [67]:
```
1 l=[6,7,8,9]
```

In [68]:
```
1 l.pop(-1)
```

Out[68]: 9

In [69]:
```
1 l
```

Out[69]: [6, 7, 8]

In [70]:
```
1 l.pop(0)
```

Out[70]: 6

In [71]:
```
1 l
```

Out[71]: [7, 8]

In [72]:
```
1 l.pop(3)
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-72-d41c6278aa32> in <module>
----> 1 l.pop(3)

IndexError: pop index out of range
```

clear method

In [73]:
```
1 l=[1,2,3,4,5,6]
```

In [74]:
```
1 l.clear??
2
```

In [75]:
```
1 l
2
```

Out[75]: [1, 2, 3, 4, 5, 6]

In [76]:
```
1 l.clear()
```

In [77]:
```
1 l
```

Out[77]: []

In [78]:
```
1 l.clear()
```

In [79]:
```
1 l
```

Out[79]: []

membership oprators

In [80]:
```
1 6 in [7,8,9]
2 7 in [7,8,9]
```

Out[80]: True

In [81]:
```
1 5 in [1,2,3,4,[5,6,]]
```

Out[81]: False

In [82]:
```
1 [5] in [1,2,3,4,[5,6]]
```

Out[82]: False

In [83]:
```
1 [5,6] in [1,2,3,4,[5,6]]
```

Out[83]: True

count

In [84]:
```
1 l=[3,3,2,3,4,5,6]
```

In [85]:
```
1 l.count??
```

In [86]:
```
1 l.count(3)
```

Out[86]: 3

In [87]:
```
1 l.count(2)
```

Out[87]: 1

In [88]:
```
1 l.count(0)
```

Out[88]: 0

index

In [89]:
```
1 l=[3,3,2,3,4,5,6]
2 l.index(3)
```

Out[89]: 0

In [90]:
```
1 l.index(3,3)
```

Out[90]: 3

reverse

In [91]:
```
1 l=[3,3,2,3,4,5,6]
```

In [92]:
```
1 l.reverse()
```

In [93]:
```
1 l
```

Out[93]: [6, 5, 4, 3, 2, 3, 3]

In [94]:
```
1 l[::-1]
```

Out[94]: [3, 3, 2, 3, 4, 5, 6]

In [95]:
```
1 l
```

Out[95]: [6, 5, 4, 3, 2, 3, 3]

In [96]:
```
1 s='python'
2 s[::-1]
```

Out[96]: 'nohtyp'

In [97]:
```
1 s
```

Out[97]: 'python'

sort

In [98]:
```
1 l=[3,3,2,3,4,5,6]
2 l.sort()
3 l
```

Out[98]: [2, 3, 3, 3, 4, 5, 6]

In [99]:
```
1 l=[3,3,2,3,4,5,6]
2 sorted(l)
3 l
```

Out[99]: [3, 3, 2, 3, 4, 5, 6]

In [100]:
```
1 l=[3,3,2,3,4,5,6]
2 print(l.sort())
```

None

universal method

- len

- min
- max
- sum
- sorted

In [101]:
```
l=[3,4,5,6,1,2,4]
print(l)
print(min(l))
print(max(l))
print(sum(l))
print(sorted(l))
```

```
[3, 4, 5, 6, 1, 2, 4]
1
6
25
[1, 2, 3, 4, 4, 5, 6]
```

In [102]:
```
l=['hii','hello','python','java']
```

In [103]:
```
sorted(l,key=len)
```

Out[103]:
```
['hii', 'java', 'hello', 'python']
```

In [104]:
```
l=['hii','hello','python','java',3]
sorted(l,key=len)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-104-486ca52d6084> in <module>
      1 l=['hii','hello','python','java',3]
----> 2 sorted(l,key=len)

TypeError: object of type 'int' has no len()
```

using key in sorted

In [105]:
```
def getlast(s):
    return s[len(s)-1]
```

In [106]:
```
m=('bb','cd','aa','zc')
```

In [107]:
```
ascode=sorted(m,key=getlast)
print(ascode)
```

```
['aa', 'bb', 'zc', 'cd']
```

iterating through list

In [108]:
```
l=[4,5,6,7]
for i in l:
    print(i)
```

```
4
5
6
7
```

In [109]:
```
for i,j in enumerate(l):
    print(i,j)
```

```
0 4
1 5
2 6
3 7
```

In [110]:
```
help()
```

```
Welcome to Python 3.8's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at https://docs.python.org/3.8/tutorial/. (ht
tps://docs.python.org/3.8/tutorial/.)

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help>

You are now leaving help and returning to the Python interpreter.
If you want to ask for help on a particular object directly from the
interpreter, you can type "help(object)".  Executing "help('string')"
has the same effect as typing a particular string at the help> prompt.
```

list comprehension

In [1]:
```
l=(x**3 for x in [1,2,3,4,5])    # example of jenrator
print(l)
```

```
<generator object <genexpr> at 0x000001C1AD174820>
```

In [2]:
```
l=[x**3 for x in [1,2,3,4,5]]    # example of jenrator
print(l)
```

```
[1, 8, 27, 64, 125]
```

WAP to return a list of even positive numbers from a given it

- input =[-5,-8,-4,0,2,3,4,6,7,9]
- output=[2,4,6]

In [113]:
```
l=[-5,-8,-4,0,2,3,4,6,7,9]
m=[]
for i in l:
    if i>0:
        if i%2==0:
            m.append(i)
m
```

Out[113]: [2, 4, 6]

In [115]:
```
l= [x for x in[-5,-8,-4,0,2,3,4,6,7,9] if x%2==0 and x>0]
```

In [116]:
```
l

```

Out[116]: [2, 4, 6]

wap to swap first and last number of a given list

- input-[5,6,7,8,9]
- output-[9,6,7,8,5]

In [6]:
```
l=[5,6,7,8,9]
l[0],l[-1]=l[-1],l[0]
```

In [7]:
```
l
```

Out[7]: [9, 6, 7, 8, 5]

wap find the sum of the member of a given list

In [10]:
```
l=[-2,-4,4,2,1,2,3]
sum=0
for i in l:
    sum+=i
sum
```

Out[10]: 6

wap which will return the sum of numbers in the list , returning 0 if the list is empty , except the number 13 is very unlucky , so it is

In [15]:
```
l=[13,1,2,3,13]
sum=0
if len(l)==0:
    sum+=0
else:
    for i in  range(len(l)):
        if l[i]==13 or l[i-1]==13  and i!=0:
            continue
        else:
            sum+=l[i]
sum
```

Out[15]: 5

wap to find the number in alist that are greter than 10 and have odd first and last digit

- input=[1,3,79,10,4,1,39,62]
- output=[79,39]
- input=[11,31,77,93,48,1,57]
- output=[11,31,77,93,57]

In [2]:
```
l=[1,3,79,10,4,1,39,62]
ans=[]
for i in l:
    if i>10 :
        if (int(str(i)[0])%2!=0) and (int(str(i)[-1])%2!=0):
            ans.append(i)
ans
```

Out[2]: [79, 39]

pb 395

Given a list L of size N, You need to count the number of special elements in the given list. An element is special if removal of that element makes the list balanced. The list will be balanced if sum of even index elements is equal to the sum of odd index elements. Example Input Input 1: A = [2,1,6,4] Input 2: A=[5,5,2,5,8] Example Output Output 1: 1 Output 2: 2 Explanation 1 : After deleting 1 from list : [2,6,4] (2+4) = (6) Hence 1 is the only special element, so count is 1. Explanation 2 : If we delete A[0] or A[1], list will be balanced (5+5)= (2+8) So A[0] and A[1] are special elements, so count is 2.

In [5]:
```
l=eval(input("enter the list ="))
print("original list",l)
count=0
for i in range(len(l)):
    c=l.copy()
    c.pop(i)
    sum1=sum(c[0::2])
    sum2=sum(c[1::2])
    if sum1==sum2:
        print("index remove",i)
        count+=1
        print("list after removing element",i,"is",l[:i]+l[i+1:])
print("total special ",count)

```

```
enter the list =[5,5,2,5,8]
original list [5, 5, 2, 5, 8]
index remove 0
list after removing element 0 is [5, 2, 5, 8]
index remove 1
list after removing element 1 is [5, 2, 5, 8]
total special  2
```

given unsorted list of integers , find the length of the longest consecutive element sequence

In [1]:
```
l=[1,3,4,6,5]
l.sort()
count=l[0]
length=0
for i in range(len(l)):
    if count==l[i]:
        length+=1
    count+=1
length
```

Out[1]: 1

dictionary

- otherwise called maping or associative arrays
- mutable
- as its a maping has no meaning
- keys are unique and can not dublicated
- keys are immutabale

creating dictionary

In [2]:
```
d={}
type(d)
```

Out[2]: dict

In [3]:
```
1 d={5}
2 type(d)
```

Out[3]:
```
set
```

In [4]:
```
1 d={'name':'python','age':50}
2 d
```

Out[4]:
```
{'name': 'python', 'age': 50}
```

In [5]:
```
1 d={(1,2,3):100,(90,100):200}
2 d
```

Out[5]:
```
{(1, 2, 3): 100, (90, 100): 200}
```

In [6]:
```
1 d={[1,2]:40}
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-6-d191d5932e4e> in <module>
----> 1 d={[1,2]:40}

TypeError: unhashable type: 'list'
```

2D ditionary also called json(java script objct notation)

In [11]:
```
1 s={'name':'python','college':'ljku','sem':3,'subject':{'dsa':50,'maths
```

In [13]:
```
1 print(s)
```

```
{'name': 'python', 'college': 'ljku', 'sem': 3, 'subject': {'dsa': 50, 'ma
ths ': 67}}
```

another method to create dictionary

In [17]:
```
1 d1=dict([('name',"python"),('city','Ahmedabad '),(3,3)])
```

In [24]:
```
1 print(d1['name'])
```

```
python
```

accessing element of dictionary using key

In [21]:
```
1 d={1:'python',2:'java',3:'abc'}
```

In [22]:
```
1 d[1]
```

Out[22]: 'python'

In [25]:
```
1 d[0]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-25-123a9cc6df61> in <module>
----> 1 d[0]

KeyError: 0
```

In [26]:
```
1 d[3]
```

Out[26]: 'abc'

using get

In [27]:
```
1 d.get(1)
```

Out[27]: 'python'

In [30]:
```
1 print(d.get(4))
```

```
None
```

In [29]:
```
1 d.get(3)
```

Out[29]: 'abc'

In [31]:
```
1 print(d.get(4,100))
```

```
100
```

looping through dictionary print key by default

In [34]:
```
1 for i in d:
2     print(i)
```

```
1
2
3
```

changing and adding item

In [35]:
```
1 d={'name':'python','age':50}
```

In [36]:
```
1 d['city']='surat'
```

In [37]:
```
1 d
```

Out[37]: {'name': 'python', 'age': 50, 'city': 'surat'}

In [38]:
```
1 d[0]=100
```

In [39]:
```
1 d
```

Out[39]: {'name': 'python', 'age': 50, 'city': 'surat', 0: 100}

In [42]:
```
1 d['name']='java'# replace name
```

In [43]:
```
1 d
```

Out[43]: {'name': 'java', 'age': 50, 'city': 'surat', 0: 100}

removing element from dictionary

In [45]:
```
1 s={1:1,2:4,3:9,4:16,5:25}
2 s
```

Out[45]: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

In [47]:
```
1 s.items()
```

Out[47]: dict_items([(1, 1), (2, 4), (3, 9), (4, 16), (5, 25)])

In [48]:
```
1 for i in s.items():
2     print(i)
```

```
(1, 1)
(2, 4)
(3, 9)
(4, 16)
(5, 25)
```

In [49]:
```
1 s.keys()
```

Out[49]: dict_keys([1, 2, 3, 4, 5])

In [51]:
```
1 s.values()
```

Out[51]: dict_values([1, 4, 9, 16, 25])

In [62]:
```
1 for i,j in s.items():
2     print(i,j)
```

```
1 1
2 4
3 9
4 16
5 25
```

pop method

In [63]:
```
print(d)
```

```
{'name': 'java', 'age': 50, 'city': 'surat', 0: 100}
```

In [64]:
```
d.pop??
```

In [65]:
```
d.pop('age')
```

Out[65]:
```
50
```

In [66]:
```
d
```

Out[66]:
```
{'name': 'java', 'city': 'surat', 0: 100}
```

In [68]:
```
d.pop('age','notfound')
```

Out[68]:
```
'notfound'
```

In [69]:
```
d
```

Out[69]:
```
{'name': 'java', 'city': 'surat', 0: 100}
```

In [70]:
```
d.pop(100)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-70-5f04a306a10f> in <module>
----> 1 d.pop(100)

KeyError: 100
```

pop item last item

In [71]:
```
d
```

Out[71]:
```
{'name': 'java', 'city': 'surat', 0: 100}
```

In [72]:
```
d.popitem()
```

Out[72]:
```
(0, 100)
```

In [73]:
```
d
```

Out[73]:
```
{'name': 'java', 'city': 'surat'}
```

In [74]:
```
d.popitem()
```

Out[74]:
```
('city', 'surat')
```

In [75]:
```
1 d
```

Out[75]:
```
{'name': 'java'}
```

In [76]:
```
1 d.popitem()
```

Out[76]:
```
('name', 'java')
```

In [77]:
```
1 d.popitem()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-77-83c64cff336b> in <module>
----> 1 d.popitem()

KeyError: 'popitem(): dictionary is empty'
```

In [78]:
```
1 d
```

Out[78]:
```
{}
```

clear

In [79]:
```
1 s
```

Out[79]:
```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

In [80]:
```
1 s.clear()
```

In [81]:
```
1 s
```

Out[81]:
```
{}
```

delete

In [82]:
```
1 del s
```

In [83]:
```
1 s
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-83-ded5ba42480f> in <module>
----> 1 s

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

In [85]:
```
s={1:1,2:4,3:9,4:16,5:25}

```

In [86]:
```
del s[2]
```

In [87]:
```
s
```

Out[87]:
```
{1: 1, 3: 9, 4: 16, 5: 25}
```

len-it would eqaul to total number of item in dictionry

In [89]:
```
s={1:1,2:4,3:9,4:16,5:25}
print(len(s))
```

```
5
```

sorted

- itsort the element of an itreble in ascending order , in dictionry it sort the keys

In [90]:
```
s={1:1,2:4,3:9,4:16,5:25}

```

In [91]:
```
sorted(s)
```

Out[91]:
```
[1, 2, 3, 4, 5]
```

In [92]:
```
s
```

Out[92]:
```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

In [93]:
```
sorted(s,reverse=True)
```

Out[93]:
```
[5, 4, 3, 2, 1]
```

In [94]:
```
s
```

Out[94]:
```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

From key

In [95]:
```
s.fromkeys('abc','def')
```

Out[95]:
```
{'a': 'def', 'b': 'def', 'c': 'def'}
```

In [96]:
```
l=[1,2,3,4,5]
s.fromkeys(l)
```

Out[96]:
```
{1: None, 2: None, 3: None, 4: None, 5: None}
```

In [98]:
```
1 s.fromkeys(l,60)
```

Out[98]: {1: 60, 2: 60, 3: 60, 4: 60, 5: 60}

In [101]:
```
1 s.fromkeys(l,65)
```

Out[101]: {1: 65, 2: 65, 3: 65, 4: 65, 5: 65}

setdefult

In [103]:
```
1 s.setdefault??
```

In [104]:
```
1 s
```

Out[104]: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

In [105]:
```
1 s.setdefault(6)
```

In [106]:
```
1 s
```

Out[106]: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: None}

In [109]:
```
1 s.setdefault(7,'abc')
```

Out[109]: 'abc'

In [110]:
```
1 s
```

Out[110]: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: None, 7: 'abc'}

In [111]:
```
1 s.setdefault(6,'fghf')
```

In [112]:
```
1 s
2
```

Out[112]: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: None, 7: 'abc'}

In [113]:
```
1 s.setdefault(7,'ac')
```

Out[113]: 'abc'

In [114]:
```
1 s
```

Out[114]: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: None, 7: 'abc'}

In [115]:
```
1 s.setdefault(8)
```

In [116]:
```
1 s
```

Out[116]: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: None, 7: 'abc', 8: None}

In [117]:
```
1 s.setdefault(8,34)
```

In [118]:
```
1 s
```

Out[118]:
```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: None, 7: 'abc', 8: None}
```

In [119]:
```
1 print(s.setdefault(8,34))
```

```
None
```

update

** syntax **

- dict.update(dict2)

In [120]:
```
1 s
```

Out[120]:
```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: None, 7: 'abc', 8: None}
```

In [121]:
```
1 s1={9:'java',10:'pycharm'}
```

In [124]:
```
1 s.update(s1)
```

In [125]:
```
1 s
```

Out[125]:
```
{1: 1,
 2: 4,
 3: 9,
 4: 16,
 5: 25,
 6: None,
 7: 'abc',
 8: None,
 9: 'java',
 10: 'pycharm'}
```

In [126]:
```
1 s2={9:'java++',10:'pycharm++'}
```

In [127]:
```
1 s.update(s2)
```

In [128]:
```
1 s
```

Out[128]:
```
{1: 1,
 2: 4,
 3: 9,
 4: 16,
 5: 25,
 6: None,
 7: 'abc',
 8: None,
 9: 'java++',
 10: 'pycharm++'}
```

copy

In [129]:
```
1 s
```

Out[129]:
```
{1: 1,
 2: 4,
 3: 9,
 4: 16,
 5: 25,
 6: None,
 7: 'abc',
 8: None,
 9: 'java++',
 10: 'pycharm++'}
```

In [130]:
```
1 s1=s.copy()
```

In [131]:
```
1 s1
```

Out[131]:
```
{1: 1,
 2: 4,
 3: 9,
 4: 16,
 5: 25,
 6: None,
 7: 'abc',
 8: None,
 9: 'java++',
 10: 'pycharm++'}
```

In [132]:
```
1 print(id(s))
2 print(id(s1))
```

```
2218642618496
2218644674816
```

dictionory comprehension

- it is a concise way to create a dictinory based on certain condition elegent way
- it consists of a key value pair followed by the for and if condition within culry braces
- **syntax***
- python {itarable:expresssion for iterator in itreable if }

In [133]:
```
1 sqaur={x:x*x for x in range(6)}
2 print(sqaur)
```

```
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

using if condition in comprehension

In [134]:
```
d={x:x*x for x in range(11) if x%2==0}
print(d)
```

```
{0: 0, 2: 4, 4: 16, 6: 36, 8: 64, 10: 100}
```

In [135]:
```
dis={'delhi':1000,'mumbai':3000,'surat':5000}
```

In [137]:
```
print({key:value*0.62 for(key,value) in dis.items()})
```

```
{'delhi': 620.0, 'mumbai': 1860.0, 'surat': 3100.0}
```

membership test based on key

In [138]:
```
d
```

Out[138]:
```
{0: 0, 2: 4, 4: 16, 6: 36, 8: 64, 10: 100}
```

In [139]:
```
3 in d
```

Out[139]:
```
False
```

In [140]:
```
10 in d
```

Out[140]:
```
True
```

wrt to merge to dic

In [141]:
```
d1={1:10,'d':'d','d':'b','d':'m'}
d2={7:77,'d':'p'}
```

In [142]:
```
d1
```

Out[142]:
```
{1: 10, 'd': 'm'}
```

In [143]:
```
d2
```

Out[143]:
```
{7: 77, 'd': 'p'}
```

In [144]:
```
d1.update(d2)
```

In [145]:
```
d1
```

Out[145]:
```
{1: 10, 'd': 'p', 7: 77}
```

for a given sentance input- the quick brown fox jumps over the lazy dog cretae the frequency mapping of a words

- oputput ={'the':2,'quick':1,.....}

In [159]:
```
s="the quick brown fox jumps over the lazy dog cretae  the frequency ma
l=s.split()
d={}
for i in l:
    d[i]=d.get(i,0)+1
d
```

Out[159]:
```
{'the': 3,
 'quick': 1,
 'brown': 1,
 'fox': 1,
 'jumps': 1,
 'over': 1,
 'lazy': 1,
 'dog': 1,
 'cretae': 1,
 'frequency': 1,
 'mapping': 1,
 'of': 1,
 'a': 1,
 'words': 1}
```

In [160]:
```
print(d)
```

```
{'the': 3, 'quick': 1, 'brown': 1, 'fox': 1, 'jumps': 1, 'over': 1, 'laz
y': 1, 'dog': 1, 'cretae': 1, 'frequency': 1, 'mapping': 1, 'of': 1, 'a':
1, 'words': 1}
```

In [161]:
```
d.get??

```

449

Write Python Program to create a dictionary with the key as the first character and value as a list of words starting with that character. Example: Input: Don’t wait for your feelings to change to take the action. Take the action and your feelings will change Output: {'D': ['Don’t'], 'w': ['wait', 'will'], 'f': ['for', 'feelings', 'feelings'], 'y': ['your', 'your'], 't': ['to', 'to', 'take', 'the', 'the'], 'c': ['change', 'change'], 'a': ['action.', 'action', 'and'], 'T': ['Take']}

In [164]:
```
s='Don’t wait for your feelings to change to take the action. Take the a
l=s.split()
d={}
for i in l:
    d[i[0]]=d.get(i[0],[])
    d[i[0]].append(i)
print(d)
```

```
{'D': ['Don’t'], 'w': ['wait', 'will'], 'f': ['for', 'feelings', 'feeling
s'], 'y': ['your', 'your'], 't': ['to', 'to', 'take', 'the', 'the'], 'c':
['change', 'change'], 'a': ['action.', 'action', 'and'], 'T': ['Take']}
```

wrt to python program to get the single digit in number sorted backword and converted to elnglish word skip the zero

- input =[1,3,4,5,11]
- output=['five','four','three','one']

In [176]:
```
l = [1, 3, 4, 5, 11]
ie = []
d={ 1: 'one', 2: 'two', 3: 'three', 4: 'four', 5: 'five',
    6: 'six', 7: 'seven', 8: 'eight', 9: 'nine'}
l.sort(reverse =True)
for i n l:
    if i < 10:
        e.append(d.get(i))
e
```

Out[176]:
```
['five', 'four', 'three', 'one']
```

count the number of 0 enclosed in 1 a given list of binary

- input=11001000100
- out put=5

In [181]:
```
l=11001000100
s=str(l)
d=[]
```

In [182]:
```
for i in s:
    d.append(i)
d
```

Out[182]:
```
['1', '1', '0', '0', '1', '0', '0', '0', '1', '0', '0']
```

In [ ]:
```

```