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 las
t)
<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 las
t)
<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]:
```
print(l)
```
```
[1, 2, 3, 4, 5, 6, 7]
```

In [26]:
```
l[4:2:-1] # l=[1,2,3,4]
```
Out[26]:
```
[5, 4]
```

List is slow And Dyanamic

In [27]:
```
l=[3,4,5,6] # list is dynamic
print(id(l[0]))
```
```
140726725846880
```

In [28]:
```
l=[3,4,5,6]
print(id(l))
```
```
1931344349696
```

In [29]:
```
print(id(l[1]))
```
```
140726725846912
```

In [30]:
```
print(id(l[2]))
```
```
140726725846944
```

list dimensions

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

In [32]:
```
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 las
t)
<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 las
t)
<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 las
t)
<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]: l=[6,7,8,9]
```

```
In [68]: l.pop(-1)
Out[68]: 9
```

```
In [69]: l
Out[69]: [6, 7, 8]
```

```
In [70]: l.pop(0)
Out[70]: 6
```

```
In [71]: l
Out[71]: [7, 8]
```

```
In [72]: l.pop(3)
---------------------------------------------------------------------------
-
IndexError                                Traceback (most recent call las
t)
<ipython-input-72-d41c6278aa32> in <module>
----> 1 l.pop(3)

IndexError: pop index out of range
```

clear method

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

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

```

```
In [75]: l

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

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

```
In [77]: l
Out[77]: []
```

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

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

```
Out[79]: []
```

membership oprators

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

```
Out[80]: True
```

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

```
Out[81]: False
```

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

```
Out[82]: False
```

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

```
Out[83]: True
```

count

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

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

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

```
Out[86]: 3
```

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

```
Out[87]: 1
```

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

```
Out[88]: 0
```

index

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

```
Out[89]: 0
```

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

reverse

In [91]:
```
l=[3,3,2,3,4,5,6]
```
In [92]:
```
l.reverse()
```
In [93]:
```
l
```
Out[93]:
```
[6, 5, 4, 3, 2, 3, 3]
```
In [94]:
```
l[::-1]
```
Out[94]:
```
[3, 3, 2, 3, 4, 5, 6]
```
In [95]:
```
l
```
Out[95]:
```
[6, 5, 4, 3, 2, 3, 3]
```
In [96]:
```
s='python'
s[::-1]
```
Out[96]:
```
'nohtyp'
```
In [97]:
```
s
```
Out[97]:
```
'python'
```

sort

In [98]:
```
l=[3,3,2,3,4,5,6]
l.sort()
l
```
Out[98]:
```
[2, 3, 3, 3, 4, 5, 6]
```
In [99]:
```
l=[3,3,2,3,4,5,6]
sorted(l)
l
```
Out[99]:
```
[3, 3, 2, 3, 4, 5, 6]
```
In [100]:
```
l=[3,3,2,3,4,5,6]
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 las
t)
<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]:
```python
l=[4,5,6,7]
for i in l:
    print(i)
```

```
4
5
6
7
```

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

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

In [110]:
```python
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]:
```python
l=(x**3 for x in [1,2,3,4,5])     # example of jenrator
print(l)
```

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

In [2]:
```python
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]
```

In [ ]:
```

```