represent type
len
max
min
sorted
reversed
Concatenation
in
index
for loop iteration
mutable-immutable
slice
Methods

```
str1=''
```

In [ ]: In [1]:

```
l1=[1,2,3,4]
l1
```

How to read
How many ways we can

Out[1]:

```
[1, 2, 3, 4]
type(l1)
```

In [2]:

Out[2]:

```
list
l2=['Apple','Banana
','Cherry'] l2
```

In [3]:

Out[3]:

```
['Apple', 'Banana', 'Cherry']
l3=[1,2,3,'A','
B','C'] l3
```

In [4]:

Out[4]:

```
[1, 2, 3, 'A', 'B', 'C']
l4=[1,2,3,'A','B','C',1
0.5,True,20+30j] l4
```

In [5]:

Out[5]:

```
[1, 2, 3, 'A', 'B', 'C', 10.5, True, (20+30j)]
l5=[100,100,
100] l5
```

In [6]:

Out[6]:

```
[100, 100, 100]
l6=[1,2,3,['A','
B','C']] l6
```

In [7]:

Out[7]:

```
[1, 2, 3, ['A', 'B', 'C']]
```

In [ ]: In [17]:

```
erry']
l3=[1,2,3,'A','B','C']
l4=[1,2,3,'A','B','C',10
.5,True,20+30j]
l5=[100,100,100]
l6=[1,2,3,['A','B','C']]
```

```
l1=[1,2,3,4]
l2=['Apple','Banana','Ch
```

```
l8=[_]
l8
```

```
Out[17]: [[[[[[[['%', '^']]]]]]]]
         l9=[_]
         l9
In
[18]:

Out[18]: [[[[[[[[['%', '^']]]]]]]]]
```

List is array of elements
That elements can be any data type
Heterogeneous
List allows duplicates
List can extend with any values
List in List possible
List of underscore , is nothing but list in list

```
         # len #
In       max #
[19]:    min #
l1       sum

Out[19]: [1, 2, 3, 4]
                    len(l1),max(l1),min(
                    l1),sum(l1)
In [23]:

Out[23]: (4, 4, 1, 10)
                                        [24]:
                                        l2
                                    In

Out[24]: ['Apple', 'Banana', 'Cherry']
                         len(l2),max(l2),min(l2)
                         # sum(l2) error
In [27]:

Out[27]: (3, 'Cherry', 'Apple')
                                        [28]:
                                        l3
                                    In

Out[28]: [1, 2, 3, 'A', 'B', 'C']
                    len(l3), # max min
                    sum fail
In [29]:

Out[29]: 6
                                    In     l4
                                    [30]:

Out[30]: [1, 2, 3, 'A', 'B', 'C', 10.5, True, (20+30j)]
                   len(l4), # max
                   min sum
In [31]:

Out[31]: 9
                                        [32]:
                                        l5
                                    In

Out[32]: [100, 100, 100]
                    len(l5),max(l5),min(
                    l5),sum(l5)
In [36]:

Out[36]: (3, 100, 100, 300)
```

In [37]:
```
l6
```

Out[37]: [1, 2, 3, ['A', 'B', 'C']]

In [38]:
```
len(l6)
```

Out[38]: 4

In [39]:
```
l66=[[1,2,3],[4
,5,6]] len(l66)
```

Out[39]: 2

In [40]:
```
l7
```

Out[40]: [[[[['%', '^']]]]]

In [41]:
```
sorted(l1)
# By default
```

```
reverse=False #
which means
ascending order
```

Out[41]: [1, 2, 3, 4]

In [42]:
```
# If you want to do
descending order #
```

```
reverse=True
sorted(l1,reverse=True
)
```

Out[42]: [4, 3, 2, 1]

In [43]:
```
sorted(l2
    )
```

Out[43]: ['Apple', 'Banana', 'Cherry']

In [44]:
```
sorted(l2,revers
e=True)
```

Out[44]: ['Cherry', 'Banana', 'Apple']

In [45]:
```
sorted(l3)
```

```
TypeError Traceback (most recent call las
t)
Cell In[45], line 1
----> 1 sorted(l3)

TypeError: '<' not supported between
instances of 'str' and 'int'
```

In [46]:
```
sorted(l3,reverse=True)
```

```
TypeError Traceback (most recent call las
t)
Cell In[46], line 1
----> 1 sorted(l3,reverse=True)

TypeError: '<' not supported between
instances of 'int' and 'str'
```

In [ ]: In [47]:
```
sorted(l3)
```

```
                                                    l5
#list object is not callable ===== magic
of python
```

Out[47]: [100, 100, 100]

**** ********** ****

**** *****

```
                                        In      l1
                                        [48]:
```

Out[48]: [1, 2, 3, 4]

```
                print(i)
In [50]: In
                4
                3
[51]:           2
                1
```

**** ******

**** ******

**** ******

In [52]: **** ******

```
val=reversed
(l1)
```

**** l1

```
for i in
val:
```

Out[52]: [1, 2, 3, 4]

```
                                        In      l2
                                        [53]:
```

Out[53]: ['Apple', 'Banana', 'Cherry']

```
        [54]:
        l1+l2
In
```

Out[54]: [1, 2, 3, 4, 'Apple', 'Banana', 'Cherry']

```
        [55]:
        l2+l1
In
```

Out[55]: ['Apple', 'Banana', 'Cherry', 1, 2, 3, 4]

```
            l1*3 #
            l1+l1+l1
In [56]:
```

Out[56]: [1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4]

**** ****

In [57]:
```
for i in s1:
 print(i)
```

```
p
y
t
h
o
n
```

In [59]:
```
for i in l1:
 print(i)
```

```
1
2
3
4
```

```
������
���
```

In [ ]:
```
-6 -5 -4 -3 -2
-1 p y t h o n 0
1 2 3 4 5
```

In [ ]:

In [63]:
```
-3 -2 -1 Apple
Banana Cherry 0
1 2
```

```
l2[0],l2[1],l2[2
]
```

```
s1='python'
```

Out[63]:
```
('Apple', 'Banana', 'Cherry')
```

```
l2[-3],l2[-2],
l2[-1]
```

In [64]:

Out[64]:
```
('Apple', 'Banana', 'Cherry')
```

In [67]:

In [ ]:

In [68]:

In [69]:

```
3
positive and negative of banana is 1 and
4
positive and negative of cherry is 2 and
5
```

```
for i in range(len(l2)):
 print(f" Positive index of {l2[i]} is
{i}")
 print(f" Negative index of {l2[i]} is
{i-3}")
 print(f" Positive index {i} and
Negative index of {l2[i]} is {i-3}")
```

In [72]: In [75]:

```
Positive index of apple is 0
Negative index of apple is -3
Positive index 0 and Negative index of
apple is -3
Positive index of banana is 1
Negative index of banana is -2
Positive index 1 and Negative index of
banana is -2
Positive index of cherry is 2
Negative index of cherry is -1
Positive index 2 and Negative index of
cherry is -1
```

```
for i in range(len(l2)):
 print(i,l2[i])
```

```
0 Apple
1 Banana
2 Cherry
```

```
for i in range(len(l2)):
 print(i,i-3)
```

```
0 -3
1 -2
2 -1
```

```
# Postive index of Apple is 0
# negative index of Apple is -3
# postive and negative index
```

```
l2=['apple','banana','cherry']
for i in range(len(l2)):
 j=i+len(l2)
 print(f'positive and negative of
{l2[i]} is {i} and {j}')
```

```
positive and negative of apple is 0 and
```

```
l7=[1,2,3,4]
l7[0]=100
l7
```

Out[75]: [100, 2, 3, 4]

List are mutable

strings are immutable

In [ ]:

In [80]:
```
list1=[1,2,3,4,5,'A','B','C','D','E',100,200,300,400,'Apple','Banana']
# P
list1[3:14:-2] # Np
```

Out[80]: (1, 2, 3)

In [1]:
```
list1=[1,2,3,['A','B']]

# Get the index of
```

Out[1]: 4

In [4]:
```
o1=list1[3]
o1[1]
```

Out[4]: 'B'

In [10]:
```
list1=[1,2,3,['A','B']]
```

Out[10]: 'B'

In [21]:
```
list2=[['Apple','Banana']]
len(list2)
```

Out[21]: ('Apple', 'Banana')

In [27]:
```
list3=[['A','B',[1,20]]]
```

Out[27]: 20

In [31]:
```
list4=['hyd',['A','B',[1,20]]]
```

Out[31]: 20

In [33]:
```
list5=[['Car']]
```

Out[33]: 'Car'

In [36]:
```
list6=[[['bus']]]
```

```
list1[3:-14:2] # NP start=3 step=+
last=-14-1=-15
list1[3:-14:-2] # NP ======> 4
list1[-3:-14:-2] # P
list1[-3:14:2] # NP =====> 400
list1[-3:-14:2] # Np
list1[::] # P
list1[::-1]# Reverse
```

```
-16 -15 -14 -13 -12 -11 -10 -9 -8 -7 -6 -5
-4 -3 -2 -1 1, 2, 3, 4, 5, 'A',
'B','C','D','E',100,200,300,400,'Apple','B
ana 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
```

```
'B' # Retrive 'B'
by using index
len(list1)
```

```
o1=list1[
3] o1[1]
```

```
list1[3][1]
```

```
# 1 means only 0
list2[0] # 2 0,1
list2[0][0],list2[0
][1]
```

```
1,20]]]  # one 0
len(list3)
list3[0][2][1]
```

```
',[1,20]]]
list4[1][2][1]
```

```
']]
list5[0][0]
```

```
']]]
list6[0][0][0
]
```

```
Out[36]: 'bus'
```

```
                          'Apple']]]]]]]]]
In [45]:                  list7[0][0][0][0][0][0][
list7=[[[[[[[['cherry',[  0][1][0]
```

```
Out[45]: 'Apple'
```

```
In [47]:
```

```
# apply the condition
# then print the output

for i in l1:
 if len(i)>5:
 print(i)
```

```
Mumbai
Bhopal
Chennai
```

```
l1=['hyd','mumbai','bhopal','chen
nai']
# print all uppercases
for i in l1:
 print(i.upper())
```

```
In [48]: In [50]:
HYD
MUMBAI
BHOPAL
CHENNAI
```

```
l1=['hyd','mu#mbai','bh#opal','ch
ennai']
# Retrive the elements which are
not having '#' for i in l1:
 if '#' not in i:
 print(i)
```

```
l1=['Hyd','Mumbai','Bhopal','Chen
nai']
# print the elements which are
having len of charcters >5 #
idea: Iterate through list
# print each element
```

```
hyd
chennai
```

```
In [60]:                  operation : list of words #
str1='hello is name my    max min
python'                   l1=str1.split()
# identify the max len word max(l1,key=len),
# min len of word         min(l1,key=len)
# idea: apply split
```

```
Out[60]: ('python', 'is')
```

iterable

if you able to print char or elements using for loop
list,string,tuple,dic

In [61]:
```
qns=['Who is PM of india?',
 'what is capital of india?',
 'No of states in india?',
 'who is the captain of ICT']

ans=['modi','delhi',29,'Rohit
sharma']

# you need to iterate through
qns list # qn1='who is pm of
india'
# ans=input("enter yout
answer") rohit


qns_list=['Who is PM of
```

Out[64]: 1

In [ ]:
```
modi

Modi ===== lower
MODI ===== lo
mODi ====

qns_list=['Who is PM of
india?',
```

In [64]:
```
india?',
 'what is capital of india?',
 'No of states in india?',
 'who is the captain of ICT']

ans_list=['modi','delhi','29'
,'Rohit sharma']

marks=0
for i in
range(len(qns_list)):
 print(qns_list[i])
 answer=input("enter the
answer:")  if
answer.casefold()==ans_list[i
].casefold():  marks=marks+1

marks
```

```
Who is PM of india?
enter the answer:modi
what is capital of india?
enter the answer:2
No of states in india?
enter the answer:delhi
who is the captain of ICT
enter the answer:rohi
```

In [ ]:
```
 'what is capital of
india?',
 'No of states in india?',
 'who is the captain of
ICT']

ans_list=['29','delhi','mo
di','Rohit sharma']

for i in qns:
 ans=
 if ans in ans_list

```

In [1]:
```
dir([])
```

Out[1]:
```
['__add__',
 '__class__',
 '__class_getitem__',
 '__contains__',
 '__delattr__',
 '__delitem__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__getstate__',
 '__gt__',
```

```
          '__hash__',
          '__iadd__',
          '__imul__',
          '__init__',
          '__init_subclass__',
          '__iter__',
          '__le__',
          '__len__',
          '__lt__',
          '__mul__',
          '__ne__',
          '__new__',
          '__reduce__',
          '__reduce_ex__',
          '__repr__',
          '__reversed__',
          '__rmul__',
          '__setattr__',
          '__setitem__',
          '__sizeof__',
          '__str__',
          '__subclasshook__',
          'append',
          'clear',
          'copy',
          'count',
          'extend',
          'index',
          'insert',
          'pop',
          'remove',
          'reverse',
          'sort']
In [ ]:
            'copy',
            'count',
            'extend',
            'index',
            'insert',
            'pop',
            'remove',
            'reverse',
            'sort'

            list1=[1,2,
In [2]: In
            3,4]

            ●●●●●
[4]:

            ●●●●●:


'append',   list1.clear
'clear',    () list1

Out[4]: []
```

```
In [6]:
# Intialise the list l1 #
l2=l1.copy()
# l1.clear()
#
```

```
print(l1),print(l2)
l1=['A','B','C']
l2=l1.copy()
l1.clear()
l1,l2
```

```
Out[6]: ([], ['A', 'B', 'C'])
```

```
In [8]:
l1=['A','B','C'
```

```
,1,2,3] l1
```

```
Out[8]: ['A', 'B', 'C', 1, 2, 3]
```

```
In [9]:
l1.pop()
```

```
# pop will delete the last element by default # It will return the value
```

```
Out[9]: 3
```

```
In [10]:
l1
```

```
Out[10]: ['A', 'B', 'C', 1, 2]
```

```
In [11]:
l1=['A','B','C',1
```

```
,2,3,100]
l1.pop()
```

```
Out[11]: 100
```

```
In [12]:
l1
```

```
Out[12]: ['A', 'B', 'C', 1, 2, 3]
```

```
In [13]:
l1=['A','B','C',1
```

```
,2,3,100]
l1.pop(0)
l1
```

```
Out[13]: ['B', 'C', 1, 2, 3, 100]
```

```
In [14]:
```

```
----------------------------------------
------------------------------- -
TypeError Traceback (most recent call las
t)
Cell In[14], line 2
 1 l1=['A','B','C',1,2,3,100]
----> 2 l1.pop(index=0)
 3 l1

TypeError: list.pop() takes no keyword
arguments
```

In [ ]:

```
l1.pop() # Remove the last value index=-1
l1.pop(0) # Remove first element
l1.pop(index=0) # Index=0 error
l1.pop(100) # if 100th index not there:
Index error
l1.pop('A') # type error
l1.pop([1,2]) # type error
```

In [17]:

```
l1=['A','B','C',1,2,3,100]
l1.pop(index=0)
l1
```

Out[17]: 3

In [21]:

```
l1=['A','B','C',1,2,3,100]
l1.pop(-2)
```

In [23]:

```
# str list tuple dic
```

```
----------------------------------------
------------------------------- -
TypeError Traceback (most recent call las
t)
Cell In[21], line 2
 1 l1=['A','B','C',1,2,3,100]
----> 2 l1.pop([1,2])

TypeError: 'list' object cannot be
interpreted as an integer
```

```
l1=['A','B','C',1,2,'A',3,100]
l1.remove('A')
l1
```

```
l1=['A','B','C',1,2,3,100]
l1.pop([1,2])

# only index
```

Out[23]: ['B', 'C', 1, 2, 'A', 3, 100]

In [ ]: In [28]:

```
# Draw back : we are manually
counting
# Draw back we can overcome by using:
index

#################### Remove
################################### #
Remove is taking only direct value,
no need to provide the index # Draw
back: Second occurence will not
consider
```

```
############### Pop
################################ #
manually we are counting the index of
element
# that index we are passing through
pop
# So that it is remvoing that
particular element
```

```
l1=['A','B','C',1,2,3,100,20,300,400,
500,'Apple']
# I want to delete element :3 using
pop
#l1.pop(5)
i=l1.index(400)
l1.pop(i)
l1
```

Out[28]: ['A', 'B', 'C', 1, 2, 3, 100, 20, 300, 500, 'Apple']

```
0,'A',400,500,'Apple']
l1.remove('A')
l1
```

In [29]:

```
l1=['A','B','C',1,2,3,100,20,30
```

Out[29]: ['B', 'C', 1, 2, 3, 100, 20, 300, 'A', 400, 500, 'Apple']

In [36]:

```
# A=0 A=9 A=12
i1=l1.index('B') # 0
i2=l1.index('B',i1+1)
i3=l1.index('B',i2+1)
print(i1,i2,i3)
```

1 5 12

- Clear

In [ ]: In [39]:

- Copy
- Pop
- Remove
- Index

```
del
```

```
l1=['A','B','C',1,2,'B',3,100,20,300,
'A',400,'B',500,'A','Apple']
del(l1[5])
l1
```

```
l1=['A','B','C',1,2,'B',3,100,20,300,
'A',400,'B',500,'A','Apple']
#l1.pop(<second A>)
```

Out[39]: ['A', 'B', 'C', 1, 2, 3, 100, 20, 300, 'A', 400, 'B', 500, 'A', 'Apple']

����������-����������-������

###

```
In [42]:
l1=[100,200]
l1.append(300
)
l1.append('Ap
ple') l1
```

```
Out[42]: [100, 200, 300, 'Apple']
```

```
In [44]:
l1=[]
l1.append(
1)
l1.append(
2)
l1.append(
3) l1
```

```
Out[44]: [1, 2, 3]
```

```
In [46]:
l1=['Hyd','Mumbai','Chenna
i']
# Print the elemnents
which are having len>3
l2=[]
for i in l1:
 if len(i)>3:
 l2.append(i)
l2
```

```
Out[46]: ['Mumbai', 'Chennai']
```

```
In [50]:
l1=['hyd','mumbai','chennai']
# Capitalize and save in a new list
l2=[]
for i in l1:
 l2.append(i.capitalize())
l2
```

```
Out[50]: ['Hyd', 'Mumbai', 'Chennai']
```

```
In [53]:
l1=['hyd','mu#mbai','che#
nnai'] # save the
elementsn which are
having '#' l2=[]
for i in l1:
 if '#' in i:
 l2.append(i)
l2
```

```
Out[53]: ['mu#mbai', 'che#nnai']
```

```
In [58]:
s='hello hai how are you'
# get the sum of all the
indexes of 'a' letter

# iterate it
# Extract the indexes of
'a'
# and save in a list
# Get the sum of those list
l1=[]
for i in range(len(s)):
 if s[i]=='a':
 l1.append(i)

sum(l1)
```

```
Out[58]: 21
```

```
In [59]:
# WAP ask the user get
10 random numbers # even
list odd list
# append even values in
even list # odd values
in odd list
import random
even_list,odd_list=[],[]
for i in range(10):

num=random.randint(20,80
)
 if num%2==0:
 even_list.append(num)
```

```
 else:                    even_list,odd_list
 odd_list.append(num)
```

```
Out[59]: ([48, 80, 72], [59, 41, 75, 65, 57, 35, 43])
```

```
         ❖❖❖❖❖❖❖❖❖❖❖❖❖
In [60]: In [61]: In 'C'] ['A', 'B', 'C']
```

```
                    l1=[1,2,3]
                    l2=['A','B','C']
                    l2.extend(l1)
                    print(l1)
                    print(l2)

                    [1, 2, 3]
[63]:               ['A', 'B', 'C', 1,
                    2, 3]


                    l1=[1,2,3]
                    l2=['A','B','C']
                    print(l1+l2) #
                    l1.extend(l2)
                    print(l1)
                    print(l2)

                    [1, 2, 3, 'A', 'B',
In [68]:            'C'] [1, 2, 3]
l1=[1,2,3]          ['A', 'B', 'C']
l2=['A','B','C']
l1.extend(l2)       l1=[1,2,3]
print(l1)           l2=[1,2,3]
print(l2)           l1.append(l2)
                    l1
[1, 2, 3, 'A', 'B',
```

```
Out[68]: [1, 2, 3, [1, 2, 3]]
                  reversed - sort

 In [ ]:
 - reverse vs     vs sorted
```

```
 In [1]: dir([])

 Out[1]: ['__add__',
          '__class__',
          '__class_getitem__',
          '__contains__',
          '__delattr__',
          '__delitem__',
          '__dir__',
          '__doc__',
          '__eq__',
          '__format__',
          '__ge__',
          '__getattribute__',
          '__getitem__',
          '__getstate__',
          '__gt__',
          '__hash__',
          '__iadd__',
          '__imul__',
          '__init__',
```

```
'__init_subclass__',
'__iter__',
'__le__',
'__len__',
'__lt__',
'__mul__',
'__ne__',
'__new__',
'__reduce__',
'__reduce_ex__',
'__repr__',
'__reversed__',
'__rmul__',
'__setattr__',
'__setitem__',
'__sizeof__',
'__str__',
'__subclasshook__',
'append',
'clear',
'copy',
'count',
'extend',
'index',
'insert',
'pop',
'remove',
'reverse',
'sort']
```

```
          ''''''''''
             'A']
             l1.count('A'
In [2]:      )
l1=['A','A',
```

```
Out[2]: 3
In [3]:
```

```
In [6]:                    l=s1.split()
s1='can can you canner can  l
not can you buy can'
```

```
 Out[6]: ['can', 'can', 'you', 'canner', 'can', 'not', 'can', 'you', 'buy', 'can']
```

```
In [10]: In [13]:
```

```
                          In [15]:
                          l1=[]
                          for word in l:
                           if word not in l1:
```

```
 print(word,":",l.count(word))
 l1.append(word)

can : 5
you : 2
canner : 1
not : 1
buy : 1
```

```
# can can not in [] True 5
['can'] # can can not in
['can'] False xxxxxxxx # you
you not in ['can'] True 2
['can','you']

can 5
can 5
you 2
canner 1
can 5
not 1
can 5
you 2
buy 1
can 5
```

```
s1='can can you canner can not
can you buy can'
list_words=s1.split()
l1=[]
for word in list_words:
 if word not in l1:

print(word,list_words.count(wo
rd))  l1.append(word)

# step1
```

```
l1=[1,2,3,4,100]
l1.insert(3,'A')
l1
# 1 2 3 'A' 4 100
```

Out[15]: [1, 2, 3, 'A', 4, 100]

In [17]:

```
l1=['hyd','Mumbai','Chennai'
,'Bengaluru'] #
l2=['Mumbai','Chennai','Beng
aluru'] l2=[]
```

In [19]:

```
for i in l1:
 if len(i)>5:
 l2.append(i)


# create the list of squares
of 1 to 10 numbers num=[]
for i in range(1,11):
 num.append(i*i)
num
```

Out[19]: [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

In [ ]:

```
for i in range(1,11):
 num.append(i*i)
num


num=[<output> <forloop
with out :>]
```

In [ ]:

In [20]:

```
num=[]
```

```
num=[i*i for i in
range(1,11)] num
```

```
Out[20]: [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
                    l2.append(i.capitali
                    ze())
In [22]:
#
l1=['hyd','mumbai']
# l2=[]
# for i in l1:
#
                    l1=['hyd','mumbai']
                    l2=[i.capitalize()
                    for i in l1] l2

Out[22]: ['Hyd', 'Mumbai']
```

������������ – 2

for and if condition

```
                    l2=[<output> <forloop>
                    <if condition>]
 In [ ]:
 l1=['hyd','Mumbai','Chen
 nai','Bengaluru'] l2=[]   l1=['hyd','Mumbai','Chen
 for i in l1:              nai','Bengaluru'] l2=[i
  if len(i)>5:             for i in l1 if len(i)>5]
  l2.append(i)             l2
 In [ ]: In [23]:


Out[23]: ['Mumbai', 'Chennai', 'Bengaluru']
```

������������ – 3

:

for-if-else

```
                     l.append(f'even {i}')
                     else:
In [26]:             l.append(f'odd {i}')
#num=[<output> <forloop>]   l
#l2=[<output> <forloop> <if
condition>]

                    [<if_output> <if condition> else
                    <else_ouput> <forloop>]
l=[]
for i in range(1,10):
 if i%2==0:

Out[26]: ['odd 1',
          'even 2',
          'odd 3',
          'even 4',
          'odd 5',
          'even 6',
          'odd 7',
          'even 8',
          'odd 9']
                       [f'even {i}' if i%2==0 else f'odd
                       {i}' for i in range(1,10)]
In [27]:

Out[27]: ['odd 1',
          'even 2',
          'odd 3',
          'even 4',
          'odd 5',
          'even 6',
```

```
          'odd 7',
          'even 8',
          'odd 9']
              List
              Sat sunday ===
 In [ ]:      Tuple
 Strings
In [ ]: In [28]:
```

- type len max min sum reversed sorted

- concatenation

- **in**

- index

- difference between **in and**

index range using **for** loop -

mutable imputable

- slice

- methods

- how to read the tuple

- different ways to read tuple

```
t1=(1,2,3)
type(t1)
```

```
Out[28]: tuple
In [29]: dir(t1)

Out[29]: ['__add__',
          '__class__',
          '__class_getitem__',
          '__contains__',
          '__delattr__',
          '__dir__',
          '__doc__',
          '__eq__',
          '__format__',
          '__ge__',
          '__getattribute__',
          '__getitem__',
          '__getnewargs__',
          '__getstate__',
          '__gt__',
          '__hash__',
          '__init__',
          '__init_subclass__',
          '__iter__',
          '__le__',
          '__len__',
          '__lt__',
          '__mul__',
          '__ne__',
          '__new__',
          '__reduce__',
          '__reduce_ex__',
          '__repr__',
          '__rmul__',
```

```
          '__setattr__',
          '__sizeof__',
          '__str__',
          '__subclasshook__',
          'count',
          'index']
```

In [ ]:

In [ ]: