List [] square bracket.

group of values seperated by comma & enclosed with in the square bracket.

---

properties iof list

- list are ordered collection.["values can't be changed"].

Ex:

```
l1=[10,20,30,40,50]
print(l1)
print(type(l1))
```

o/p

```
[10, 20, 30, 40, 50]
<class 'list'>
```

- list are mutable.

Ex:

can change, remove,delete----mutable.

can't add, remove,delete-----immutable

- list support duplicate elemments.

Ex:

```
l1=[10,20,10,40,50,20,10]
print(l1) # [10, 20, 10, 40, 50, 20, 10]
```

- list can be homogenious or hetrogenious.

Ex:homogenious

```
l1=[10,20,30,40]
```

Ex:hetrogenious.

```
l2=[10,True,12.1,6+1j,"manga"]
```

- list support indexing both +ve & _ve.

Ex:

```
l1=[10,20,2+1j,"manga",True]
print(l1[0]) # 10
print(l1[1]) # 20
print(l1[2]) # (2+1j)
print(l1[3]) # manga
print(l1[4]) # True
print(l1[5]) # IndexError: list index out of range.
```

[if we try to print the index that is not in the list it gives error]

list support slicing opertator.

deriving a list from an existing list called slicing a list.append

syntax:
list[s:stop/end:step]

Ex:
```
l1=[10,20,30,40,50,60,70]
print(l1[0:])  # [10, 20, 30, 40, 50, 60, 70]
print(l1[1:5]) # [20, 30, 40, 50]
print(l1[0:7:2]) # [10, 30, 50, 70]
print(l1[:7]) # [10, 20, 30, 40, 50, 60, 70]
```
---

built in methods in list.

append, insert, extend, pop, remove, clear, copy, sort, reverse, count, index
---

1) append:
only on element added at once.
syntax:
```
l=[10,20]
l.append("kalla")
print(l)  # [10, 20, 'kalla']
l.append(["malla",420])
print(l)  # [10, 20, 'kalla', ['malla', 420]]
```
---

2) insert: index starts with 0
syntax:
list.insert(index,object)
Ex:
```
l1=[10,20,40]
l1.insert(1,420)
print(l1)  # [10, 420, 20, 40]
```
---

3) extend:
syyntax:
list.extend(iterable)---None
Ex:
```
l1=[10,20,40,30]
print(l1)
l1.extend([1,2,3,"manga"])
print(l1) # [10, 20, 40, 30, 1, 2, 3, 'manga']
```
---

4) pop: it will remove the value based on index.

syntax:

list.pop(index:-1)-----any

Ex:

```
l=[1,"manga",3,4,5,2]
l.pop(1)
print(l) # [1, 3, 4, 5, 2]
l.pop()
print(l) # [1, 3, 4, 5]
l.pop(2)
print(l) # [1, 3, 5]
```

default index value is -1, if we not tell ondex it will remove the -ve index -1 valie.

---

5) remove: remove the value directly.

one value remove at a time.

syntax:

list.remove(value)----None

Ex:

```
l=[10,20,'manga',4,2,1]
l.remove(10)
c1=l
print(c1) # [20, 'manga', 4, 2, 1]

l.remove("manga")
c2=l
print(c2) # [20, 4, 2, 1]

l.remove()  # need to give one value for removing. IMP
c3=l
print(c3)  # TypeError: list.remove() takes exactly one argument (0 given)

l.remove(10,20)  # TypeError: list.remove() takes exactly one argument (2 given). IMP
c3=l
print(c3)

l.remove(100) # list remove the values which is present in the list only.
c5=l
print(c5) # ValueError: list.remove(x): x not in list.

""" # In list, the remove() method always removes the FIRST matching value.

# ✓ Example
# numbers = [10, 20, 30, 20, 40]
# numbers.remove(20)
```

```
# print(numbers)

# ✓ Output
# [10, 30, 20, 40]

# 💬 Explanation

# It removed only the first 20.

# The second 20 is still there.

# ⭐ Important Points

# remove() removes by value, not by index.

# It removes only the first occurrence.

# If value is not found → it gives error."""
```
---

6) clear: empty the list.

syntax:

list.cleara()----None

Ex:

```
l=[1,2,3,4,5]
print(l)  # [1, 2, 3, 4, 5]
l.clear()
print(l) # []  remove all the elements in the list.
```
---

7)copy: copy a list.

syntax:

list.compy()--list

Ex:

```
l=[10,2,3,4,'manga']
print(l) # [10, 2, 3, 4, 'manga']
l2=l.copy()
print(l2) # [10, 2, 3, 4, 'manga']
```
---

8) sort: sort the list. by default ascending(kammi inda jasti)

syntax:

Ex:

```
l=[1,5,4,7,9,2,3]
l.sort()
```

```
print(l) # [1, 2, 3, 4, 5, 7, 9]
 or
print(sorted(l)) # [1, 2, 3, 4, 5, 7, 9]

l.sort(reverse=True)
print(l) # [9, 7, 5, 4, 3, 2, 1]
```
---

9) reverse: reverse the complete list.

Ex:

```
l=[1,5,4,7,9,2,3]
l.reverse()
print(l)  # [3, 2, 9, 7, 4, 5, 1]
```
---

10) count(): count the perticular value repeated.

syntax:

list.count(value)-----int

Ex:

```
l=[10,20,30,10,40,50,10]
print(l.count(10)) # 3
```
---

11) index: return the index number of given value.(yava index ale ede.)

syntax:

list.index(value,start,stop)----int

Ex:

```
l1=[10,20,30,40,50]
print(l1)
print(l1.index(10,0,5)) # 0, it will give the index number of the element which we have passed.
```

Matrix:

```
m=[[1,2],[3,4]]
print(m[0])  # [1, 2]
print(m[1])  # [3, 4]
```

List comprehension

◇ List Comprehension

Used to create a list in one line.

✓ Example 1: Squares of numbers 1 to 5

```
squares = [x*x for x in range(1, 6)]
print(squares)
```

Output: [1, 4, 9, 16, 25]

✓ Example 2: Even numbers from 1 to 10

```
evens = [x for x in range(1, 11) if x % 2 == 0]
print(evens)
```

Output: [2, 4, 6, 8, 10]

Ex:

```
names=["james","jam","jangly","mamu","manga"]
j_name=[]
for name in names:
    if 'j' in name:
        j_name.append(name)
print(j_name)  # ['james', 'jam', 'jangly']
```