# List

marks = [11.6, 12.2, 13.4]

print (marks)

↓

[11.6, 12.2, 13.4]

print (type (marks))

↓

list.

*). Elements can be accessed by.

indexing. marks[0] → 11.6
marks[1] → 12.2.

print (marks[0])

↓

11.6.

---

*) List can store multiple datatype data.

Eg:- student = ["Varun", 95.8, "Bang alore"].

print(student)

↓

['Varun', 95.8, 'Bangalore']

---

*). Strings are immutable in Python.

*). Lists are mutable

Eg:- student[0] = "Man"

↓

Allowed.

*) Slicing is allowed in List

marks = [82, 85, 86, 87, 88]

0 1 2 3 4

-5 -4 -3 -2 -1

marks [1 : 3]

↓

[85, 86]

marks [-3 : -1] -> [86, 87]

---

# List methods

list = [2, 1, 3]

1]. list.append(4) [2, 1, 3, 4]

2]. list.sort() [1, 2, 3]

3]. list.sort(reverse = True) [3, 2, 1]

4]. list.reverse() -> [3,1,2]

5]. list.insert(index, element)

Eg:- list.insert(0,10)

↓

[10,1,3].

---

*) list.sort() can be used in strings also.

fruits = [Mango, guava, apple].

print(fruits.sort())

↓

[apple, guava, Mango].

---

6]. list.remove(1)

list = [2,1,3].

↓

[2,3]

incase list: [2,1,3,1]

↓ removes first 1.

[2,3,1]

7]. list.pop(index)

list = [2,1,3]

↓

list.pop(1)

[2,3]

8]. list.count(element)

↓

counts number of occurences of element.

# Tuples

tup = (87, 64, 33)

*) Indexing is only used to access elements.

*). tuples are immutable

tuple[0] = 20

// Not allowed

*). To consider it as tuple.,

adding comma after each value is necessary.

Eg:- tup = (1)

print(type(tup))

↓

class 'int'

```
tup = (1,)
print(type(tup))
```

↓

class 'tuple'.

// In case of multiple elements.
no need of comma at end.

```
tup = (1,2,3)
print(type(tup))
```

↓

class 'tuple'.

**4). Slicing is allowed in tuples.**

```
tup = (1,2,3,4)
print(tup[1:3])
```

↓

(2,3) //

# Tuple Methods

tup = (2, 1, 3, 1)

1]. tup.index(el) → returns first occurrence of element.

tup.index(1)

↓

1

=

2]. tup.count(el) → no. of occurrences.

tup.count(1)

↓

2

=