08/10/25

Tuple:

→ A tuple in python is a collection of ordered and immutable elements.

→ It is similar to a list, but you cannot modify a tuple after it is created.

→ It is represented by tuple() or { }.

→ It is also multivalued heterogeneous variable.

Syntax:

```
tuple_name = (element1, element2, element3, ...)
```

Example:

```
t1 = (34, "vasu", 27.9, 100, false, 34, 1+8j, "vasu")
print(t1)
```

o/p:

```
(34, 'vasu', 27.9, 100, false, 34, 1+8j, 'vasu')
```

Tuple Methods in python:

1. count()

Returns the number of times a specific value appears in the tuple.

Syntax:

```
tuple.count(value)
```

Ex:

```
numbers = (1,2,2,3,4,2)
print(numbers.count(2))
```

O/p:

3.

2. Index()

Returns the index (position) of the first occurence of a value in the tuple.

Syntax:

```
tuple.index(value)
```

Ex:

```
fruits = ("apple", "banana", "cherry", "apple").
print(fruits.index("apple"))
```

O/p:

0

Note :

→ Tuple don't support methods like append(), remove(), sort(), or reverse(). because they cannot be modified.

→ But you can still use built-in functions like :

* len(tuple)
* max(tuple)
* min(tuple)
* Sum(tuple) → for numeric data.
* Sorted(tuple) [returns a list, not a tuple)

Set :

→ It is a multivalued heterogeneous variable.

→ It is mutable, unordered and it cannot allow duplicates.

→ It is represented by set() or { }.

(or)

→ A set in python is a collection of unique and unordered elements.

→ It is used to store multiple items in a single variable without duplicates.

Syntax :

set_name = {element 1, element 2, element 3, ...}.

Example :

S1 = {34, "vasu", 27.9, 100, false, 34, 1+8j, "vasu", "vaishu"}.

print(S1).

O/p :

= {false, 34, 100, 27.9, (1+8j), 'vaishu', 'vasu'}.

Set Methods in python :

Sets have several built-in methods that let you add, remove, and perform operations on elements.

1. add().

→ Adds a single element to the set.

→ If the element already exits, it won't be added again.

Ex :

```
S = {1, 2, 3}
S.add(4)
print(S).
```

O/p :

= {1, 2, 3, 4}.

2. update ( )

Adds multiple elements (from list, tuple, or another set)

```
S = {1, 2}
S.update ([3,4,5])
print(S)
```

O/p : {1, 2, 3, 4, 5}

3. remove ( )

Removes an element.

→ If the element doesn't exit, it gives an error.

```
S = {1, 2, 3}
S.remove (2)
print (S)
```

O/p : {1, 3}

4. discard ( ).

Removes an element if present.

→ No error if the element doesn't exist.

```
S = {1, 2, 3}
S. discard (3)
print (S)
```

O/p :

{1, 2}

5. pop()

Removes and returns a random element from the set.

```
S = {10, 20, 30}
item = S.pop()
print(item)
print(S)
```

o/p: 10.
{20, 30}

6. clear()

Removes all elements, leaving an empty set.

```
S = {1, 2, 3}
S.clear()
print(S)
```

o/p:
set().

7. copy()

Returns a shallow copy of the set.

```
S1 = {1, 2, 3}
S2 = S1.copy()
print(S2)
```

o/p:
{1, 2, 3}

1. union() → Returns all elements from both sets.

2. intersection() → Returns common elements.

3. difference() → Elements present in first but not in second.

4. Symmetric_difference() → Elements not common to both.

5. isdisjoint() → Returns True if no common elements.

6. issubset() → checks if set is subset of another.

7. issuperset() → checks if set is superset of another.