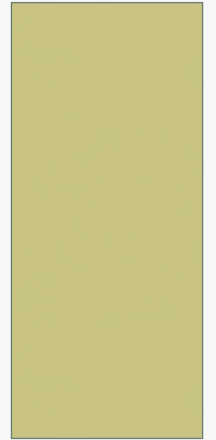


# PYTHON TUPLE

-PYTHON SERIES



# INTRODUCTION Transforming Lives Digitally

- A tuple is a collection which is ordered and **unchangeable**.
- Tuples are written with round brackets.
- It is a collection of objects which ordered and immutable.
- Allow duplicate values.

# CREATE

- #Empty Tuple

```
empty_tuple = ()  
print (empty_tuple)
```

## #Single Element

To write a tuple containing a single value you have to include a comma, even though there is only one value

```
x= (50,)
```

# CONTINUE..

- #mutiple element

```
x= ("apple", "54.23", "23", "apple", "India")
```

```
print(thistuple)
```

# TUPLE TYPECASTING

- X = 'hello'

Y = tuple(x)

Print(y)

# THE TUPLE() CONSTRUCTOR

- It is also possible to use the tuple() constructor to make a tuple.
- `x= tuple(("apple", "54.55", 23))`
- # note the double round-brackets

# READ OR ACCESS

- To access values in tuple, use the square brackets
- For multiple elements use slicing
- $X = (10,20,30)$
- $X[0]$
- $X[0:3]$  #Range of Indexes
- You can use positive and negative index both.

# UPDATE

- Tuples are unchangeable, that you cannot change, add, or remove items once the tuple is created.

- `tup1 = (12, 34.56)`

# Following action is not valid for tuples

- `tup1[0] = 100 #throw Error`



# WORKAROUND

- `tup3 = tup1 + tup2` #concatenation

#converting to list

```
x = ("apple", 54.36, 23)
```

```
y = list(x)
```

```
y[1] = "hello"
```

```
x = tuple(y)
```

# DELETE

- Tuples are **immutable**, so you cannot remove items from it.

```
del a[0] #throw error
```

```
#you can delete entire tuple
```

```
del a
```

# PACKING AND UNPACKING

- When we create a tuple, we normally assign values to it. This is called "packing" a tuple
- we are also allowed to extract the values back into variables. This is called "unpacking"

```
x= ("apple", 53.36, 23)
```

```
(a, b, c) = x
```

#The number of variables must match the number of values in the tuple

# TUPLE METHOD

- Len()
- Count() *#tuple.count(value)*
- Index() *#tuple.index(value)* raise exception if not found
  
- Concatenation
- Mutiplication

- `Max()` #It returns the item from the tuple with the highest value.
- You can compare the same datatype – in case of number it will return max value
- In case of all string it will return on the basis of highest ASCII value
- In case of combination it will raise error

- `Min` – #returns minimum value
- `sum()` - #This function returns the arithmetic sum of all the items in the tuple. Unsupported for string
- `Any()` #If even one item in the tuple has a Boolean value of `True`, then this function returns `True`. Otherwise, it returns `False`.

- `All()` #returns True only if all items have a Boolean value of True. Otherwise, it returns False.
- `Sorted()` #This function returns a sorted version of the tuple. The sorting is in ascending order

# MEMBERSHIP OPERATION

- In
- not in
  
- Syntax

'string' in/notin(Iterable object)



# NESTED TUPLES IN PYTHON

- `a=((1,2,3),(4,(5,6)))`
- `Print(a[1][1][1]) #6`