التاريخ: / /

الموضوع: _____

Functions

*Args and **kwargs

Args (Positional expansion):

```
tup = 1,2,3,4,5
a,b,*c = tup  ⟶ 1, 2, [3,4,5]
a,*b,c = tup  ⟶ 1, [2,3,4], 5
*a,b,c = tup  ⟶ [1,2,3], 4, 5
```

→ * Variable Collect Rest of items in a list

⇒ Def F(*args): Reciving Varing element
    Print(args)

F(1,2,3,4,5) ⟶ tuple of the input

F(tup) ⟶ ((1,2,3,4,5),)
↳ element of a tuple

F(*tup) ⟶ unpack it and it become Normal tuple

in lists

```
lst1 = [1,2,3]
lst2 = [4,5,6]
lst3 = [7, _ _ _ , 13]
Print(lst1) ⟶ [1,2,3]
Print(*lst1) ⟶ 1,2,3 ⟶ unpacked from list

Print([*list1, *list2]) ⟶ unpack 2 lists in one list
                          [1,2,3,4,5,6]
```

unPack items iterate over Coulmn

التاريخ: الموضوع:

Print(*zip(L1, L2, L3)) ⟶ 3 Coulmns of them

it will Stop at Shortest. (1,4,7), (2,5,8), (3,6,9)

** Kwargs

It's Similar to *arg ⟶ accept Positional argument ; But Kwargs accept Keywords.

⇒ Examples- Can Recive Variable/numbers Convert them to Dct Dict of items

def hello(**kwargs):

For key, value in kwargs.items():

key value Print(key, value)

hello(N="Ahmed", D="5/6/2004", grades=20)

It will be Printed as Dict value of a key

⇒ We can Put both in a single function:

def F(*args, **kwargs):

Print('args:', args, 'kwargs:', kwargs)

Example ⟶ it will be used based on type of input

F(1,2) ⟶ args: (1,2), kwargs: {}

F(a=10, b=20) ⟶ kwargs: {'a': 10, 'b': 20}, args empty

F(1, 2, a=10, b=20) ⟶ both will be Printed as tuple and Dct

⇒ Note: We Can't Pass Positional arg. after Keyword Argument.

F(a=10, 1) ⟶ Error

F(*args, ** kwargs) ⟶ error

Standard, Positional and Keyword arguments:-

```
Def FC(a, b, *myargs, **myKwargs):
    Print(a, b, 'args', myargs, 'Kwargs', MyKwargs)
```

FC(1,2) ⟹ 1, 2, args () Kwargs ()

FC(a=10, b=10) ⟶ 10 20 args () Kwargs ()

⇒ Note: FC(x=10, y=20) ⟶ We can't add argument keyword before Positional argument.

⇒ FC(1, 2, 3, 4, 5, x=10, y=20)
↳ 1 2 args (3, 4, 5) Kwargs {'x': 10, 'y': 11}

⇒ FC(a=1, b=2, a=10, b=20) ⟶ SyntaxError: keyword argument Repeated

---

Merging Dictionaries

⇒ **dct will expand to its tuple of (key, value), *dict ⟶ will Return (Keys)

```
dct1 = {'A': 10, 'B': 20}
dct2 = {'C': 30, 'D': 40}
Print(*dict1) ⟶ Will Return keys only ⟶ A B

dct = {**dct1, **dct2} ⟶ Will merge both Dictionaries.
Print(dct)
```