Functions

January 5, 2022

[ ]:
```
# **args
```

[13]:
```
def myfunc(a,b,):
    # returns 5 % of the sum
    return sum((a,b)) * 0.05
```

[7]:
```
myfunc(10,20)
```

[7]:
```
1.5
```

[14]:
```
# if we provide 3 arguments
myfunc(1,2,3)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_3624/732755594.py in <module>
      1 # if we provide 3 arguments
----> 2 myfunc(1,2,3)

TypeError: myfunc() takes 2 positional arguments but 3 were given
```

[22]:
```
# Now using * args we can provide any number of arguments
def myfunc(*args):
    return sum((args)) * 0.05
```

[23]:
```
myfunc(1,2,3,4,5)
```

[23]:
```
0.75
```

[24]:
```
# We can also print the args
def myfunc(*args):
   print(args)
```

[25]:
```
myfunc(10,10,20,30,40)
```

```
(10, 10, 20, 30, 40)
```

[26]:
```
def myfunc(*args):
    for item in args:
        print(item)
```

[27]:
```
myfunc(10,20,30,40)
```

```
10
20
30
40
```

[ ]:
```

```

[39]:
```
# **kwargs
def myfunc(**kwargs):
    if 'fruits' in kwargs:
        print('My favourite fruit is {}'.format(kwargs['fruits']))
    else:
        print('My fruit is not present')
```

[41]:
```
myfunc(fruits = 'apple')
```

```
My favourite fruit is apple
```

[48]:
```
def dict1(**kwargs):
    print(kwargs)
    if 'fruit' in kwargs:
        print('My favourite fruit is {}'.format((kwargs['fruit'])))
    else:
        print('Your fruit is not present')
```

[ ]:
```

```

[49]:
```
dict1(fruit = 'grapes',food = 'Idly')
```

```
{'fruit': 'grapes', 'food': 'Idly'}
My favourite fruit is grapes
```

[50]:
```
# now using *args and **kwargs together
```

[52]:
```
def myfunc(*args,**kwargs):
    print(args)
    print(kwargs)
    print('I would like to eat {} {}'.format(args[0],kwargs['food']))
```

[53]:
```
myfunc(10,20,30,food = 'idly',fruits = 'guava')
```

```
(10, 20, 30)
{'food': 'idly', 'fruits': 'guava'}
I would like to eat 10 idly
```

[54]:
```
# function that returns only even numbers
```

[65]:
```
def myfunc(*args):
    even_list = []
    for num in args:
        if num % 2 == 0:
            even_list.append(num)
    return even_list
```

[66]:
```
myfunc(10,20,34,7)
```

[66]:
```
[10, 20, 34]
```

[ ]:
```

```