In [ ]:
```
- function without arguments
- function with arguments
- Defalut
- local vs global
- return
- function in function
```

In [ ]:
```
```

In [ ]:
```
```

In [3]:
```
#Functions with out arguments
def even_odd():
    num = eval(input("Enter a number:"))
    if num%2==0:
        print('Even')
    else:
        print('Odd')

even_odd()
```

```
Enter a number:78
Even
```

In [4]:
```
#Functions with  arguments
def even_odd(num):

    if num%2==0:
        print('Even')
    else:
        print('Odd')

even_odd(78)
```

```
Even
```

In [6]:
```
#Default Arguments:this means we can fix the argument values or the argument is
#Defalut argument should always be at last i.e second parameter.
def summ(num1,num2=100):
    print("num1:", num1)
    print("num2:", num2)
    add=num1+num2
    print(add)
summ(20)
#num1 = 20 num2=100
```

```
num1: 20
num2: 100
120
```

In [8]:
```
def summ(num1=200,num2=100):
    print("num1:", num1)
    print("num2:", num2)
    add=num1+num2
    print(add)
summ()
```

```
num1: 200
num2: 100
300
```

- Note : Deafult arguments is always at last

In [ ]:
```
avg(n1,n2,n3=100) #Works
avg(n1,n2=100,n3=100)#Works
avg(n1=100,n2=100,n3=100) #Works
avg(n1=100,n2,n3)#Fails
avg(n1=100,n2=100,n3) #Fails
avg(n1=100,n2,n3=100) #Fails
avg(n1,n2=100,n3=100)#Works
```

In [10]:
```
#Case-1:
def avg(n1,n2,n3=100):
    averge=(n1+n2+n3)/3
    print(averge)
avg(200,300)  #n1=200 n2=300 n3=100
```

```
200.0
```

In [11]:
```
#Case-2:
def avg(n1,n2=100,n3=100):
    averge=(n1+n2+n3)/3
    print(averge)
avg(300)
```

```
166.66666666666666
```

In [13]:
```
#Case-3:
def avg(n1=100,n2=100,n3=100):
    averge=(n1+n2+n3)/3
    print(averge)
avg()
```

```
100.0
```

In [14]:
```
#Case-4:
def avg(n1=100,n2,n3):
    averge=(n1+n2+n3)/3
    print(averge)
avg(100,200)
```

```
  Cell In[14], line 1
    def avg(n1=100,n2,n3):
                      ^
SyntaxError: non-default argument follows default argument
```

In [17]:
```
#Case-5:
def avg(n1=100,n2=100,n3):
    averge=(n1+n2+n3)/3
    print(averge)
avg(200)
```

```
  Cell In[17], line 1
    def avg(n1=100,n2=100,n3):
                             ^
SyntaxError: non-default argument follows default argument
```

In [18]:
```
#Case-6:
def avg(n1=100,n2,n3=100):
    averge=(n1+n2+n3)/3
    print(averge)
avg(100)
```

```
  Cell In[18], line 1
    def avg(n1=100,n2,n3=100):
                      ^
SyntaxError: non-default argument follows default argument
```

In [19]:
```
#Case-7:
def summ(n1,n2=100):
    averge=n1+n2
    print(averge)
summ(100)
```

```
100.0
```

In [ ]:
```
def avg(n1,n2=100 ):
    averge=(n1+n2+n3)/3
    print(averge)
avg(100)
# have you provided any value before define the function
# what are you provided while defineing the function
# what are you provided while calling the function num2= 100
# what are you provided while calling the function num2=500
```

In [ ]:
```
#define call then kaam
```

In [20]:
```
#Create the default argument for tip_percentage
def totalbill(bill, tip=10):
#     bill = eval(input("Enter the bill:"))
#     tip =eval(input("Enter the tip percentage :"))
    tip_percent =(bill*tip)/100
    total_bill=bill+tip_percent
    print(f"The total bill is {total_bill} for the tip percent {tip}")

totalbill(1000)
```

```
The total bill is 1100.0 for the tip percent 10
```

In [ ]:
```
#wap ask the user enter salary :100000
#ask the user enter tax percentage : 10
#calculate tax
#implement fnction without argument
#function with argument
#function default argument :tax_per = 10
```

In [23]:
```
#implement fnction without argument
def salary():
    sal=eval(input('Enter the salary:'))
    tax=eval(input('Enter the tax percentage:'))
    tax_per=(sal*tax)/100
    print(f'The salary is {sal} and tax percent is {tax_per}')
salary()
```

```
Enter the salary:100000
Enter the tax percentage:10
The salary is 100000 and tax percent is 10000.0
```

In [25]:
```
#function with argument
def salary1(sal,tax):

    tax_per=(sal*tax)/100
    print(f'The salary is {sal} and tax percent is {tax_per}')
salary1(100000,10)
```

```
The salary is 100000 and tax percent is 10000.0
```

In [26]:
```
#function default argument :tax_per = 10

def salary1(sal,tax=10):

    tax_per=(sal*tax)/100
    print(f'The salary is {sal} and tax percent is {tax_per}')
salary1(100000)
```

```
The salary is 100000 and tax percent is 10000.0
```

- Return Statements

In [ ]:
```
# tax_per we re getting inside the function whereas,
# if i want to use it outside I am supposed to use
# retun function
```

In [32]:
```
def salary1(sal,tax=10):
    tax_per=(sal*tax)/100
    return(tax_per)
#we are asking function to return a value
tax_per=salary1(100000)
print(tax_per)
print(sal)
```

```
10000.0

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[32], line 7
      5 tax_per=salary1(100000)
      6 print(tax_per)
----> 7 print(sal)

NameError: name 'sal' is not defined
```

In [ ]:
```

```

In [35]:
```
#WAP take 3 numbers do the sum seperately and average seperately and return sum
def mathe(n1,n2,n3):
    summ=n1+n2+n3
    avr=(n1+n2+n3)/3
    return(avr,summ)
avr,summ=mathe(10,100,10000)
print(summ)
print(avr)
```

```
10110
3370.0
```

In [1]:
```
import random
def game1():
    n1=random.randint(1,10)
    n2=eval(input("Enter the number :"))
    if n1== n2:
        print("in")
    else:
        print("out")
game1()
```

```
Enter the number :78
out
```

In [3]:
```
import random
def game1(n2=7):
    n1=random.randint(1,10)
    #n2=eval(input("Enter the number :"))
    if n1== n2:
        print("in")
    else:
        print("out")
game1()
```

```
out
```

In [4]:
```
import random
def game1(n2=7):
    n1=random.randint(1,10)
    #n2=eval(input("Enter the number :"))
    if n1== n2:
        print("in")
        return(10000,"award")
    else:
        print("out")
        return(0,"no award")
money=game1()
money
```

```
out
```

Out[4]:
```
(0, 'no award')
```

- Local variables :
  - the variables are initialized inside the function
  - its like movie in a single state

- Global Variable:
  - The variables are initialized outside th function
  - its more like a pan india movie

In [ ]:
```
n1 = random.randint(1,10) #global
```

In [ ]:
```
num=10 #Global variable
def even_odd(num):
    num = eval(input("Enter a number:")) #local variable
    if num%2==0:
        print('Even')
    else:
        print('Odd')

even_odd(20) #local

even_odd(eval(input("Enter a number:"))) #local

num=eval(input("Enter a number:")) #global
even_odd(num)

num=random.randint(1,50) #global
even_odd(num)
#s1=define the function
#s2=call the function
```

- variables inside the function is called as local variables
- variables outside the function is called as global variables
- if you want to use variable inside the function initialize that beefore calling the function

- if you want to use local variables outside the function provide return statement

In [ ]:
```
#important function
use local variables outside the function without using return

global
```

In [5]:
```
#Create the default argument for tip_percentage
def totalbill(bill, tip=10):
    global totalbill
#     bill = eval(input("Enter the bill:"))
#     tip =eval(input("Enter the tip percentage :"))
    tip_percent =(bill*tip)/100
    total_bill=bill+tip_percent
    print(f"The total bill is {total_bill} for the tip percent {tip}")

totalbill(1000)
```

```
The total bill is 1100.0 for the tip percent 10
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[5], line 11
      8     print(f"The total bill is {total_bill} for the tip percent {ti
p}")
     10 totalbill(1000)
---> 11 print(tim)

NameError: name 'tim' is not defined
```

In [6]:
```
#wap ask take three numbers as arguments
#create add and avg variables inside the function
#calculate that add and avg
#print outside function without useing retuen function
# num1=eval(input("Enter the value for num1"))
# num2=eval(input("Enter the value for num2"))
# num3=eval(input("Enter the value for num3"))
def look(num1,num2,num3):
    global add,avg
    add=num1+num2+num3
    avg=round((num1+num2+num3)/3)
look(2,3,4)
print(add)
print(avg)
```

```
9
3
```

Functions in functions

In [7]:
```
def greet1():
    print("hello")
def greet2():
    print("How are you?")

greet1()
greet2()
```

```
hello
How are you?
```

In [8]:
```
def greet1():
    print("hello")
def greet2():
    greet1()
    print("How are you?")

greet2()
```

```
hello
How are you?
```

In [9]:
```
def greet1():
    greet2()
    print("hello")
def greet2():

    print("How are you?")

greet1()
```

```
How are you?
hello
```

In [11]:
```
try:
    def greet1():
        greet2()
        print("hello")
    def greet2():
        greet1()
        print("How are you?")
except Exception as e:
    print(e)

greet1()
```

```
---------------------------------------------------------------------------
RecursionError                            Traceback (most recent call last)
Cell In[11], line 11
      8 except Exception as e:
      9     print(e)
---> 11 greet1()

Cell In[11], line 3, in greet1()
      2 def greet1():
----> 3     greet2()
      4     print("hello")

Cell In[11], line 6, in greet2()
      5 def greet2():
----> 6     greet1()
      7     print("How are you?")

Cell In[11], line 3, in greet1()
      2 def greet1():
----> 3     greet2()
      4     print("hello")

Cell In[11], line 6, in greet2()
      5 def greet2():
----> 6     greet1()
      7     print("How are you?")

    [... skipping similar frames: greet1 at line 3 (1484 times), greet2 at li
ne 6 (1484 times)]

Cell In[11], line 3, in greet1()
      2 def greet1():
----> 3     greet2()
      4     print("hello")

Cell In[11], line 6, in greet2()
      5 def greet2():
----> 6     greet1()
      7     print("How are you?")

RecursionError: maximum recursion depth exceeded
```

In [ ]:
```
#calculator program
#create four functions
#fun1: add
#fun2 : sub
#fun3 : mul
#fun4 : div

#print("if you want to use add operation please enter 1 ")
#print("if you want to use sub operation please enter 2 ")
#print("if you want to use mul operation please enter 3 ")
#print("if you want to use div operation please enter 4 ")

#option=eval(input("choose option 1,2,3,4"))

#if option==1:
#n1=
#n2=
#add(n1,n2)
#elif option==2
#sub()
```

In [21]:
```
def add(n1,n2):
    return n1+n2
def sub(n1,n2):
    return n1-n2
def mul(n1,n2):
    return n1*n2
def div(n1,n2):
    return n1/n2
value=eval(input("Enter the number 1 for addition, 2 for subtraction, 3 for mul
n1=eval(input("Enter the value of n1: "))
n2=eval(input("Enter the value of n2: "))
if value==1:
    result=add(n1,n2)
elif value==2:
    result=sub(n1,n2)
elif value==3:
    result=mul(n1,n2)
elif value==4:
    result=div(n1,n2)
else:
    print("Please enter the right option")
print(result)
```

```
Enter the number 1 for addition, 2 for subtraction, 3 for multiplication, 4 f
or division1
Enter the value of n1: 45
Enter the value of n2: 45
90
```

In [ ]: