In [1]:
```
n=eval(input())
```

```
python
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[1], line 1
----> 1 n=eval(input())

File <string>:1

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

In [ ]:
```
if <condition>
elif <condtion>
else # No condition
```

In [2]:
```
if a==10
print('hello')
```

```
  Cell In[2], line 2
    print('hello')
    ^
IndentationError: expected an indented block after 'if' statement on line 1
```

In [4]:
```
if a==10:
    print('hello')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[4], line 1
----> 1 if a==10:
      2     print('hello')

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

In [ ]:
```
summ=10
max=25
```

**Functions**

- Reuse any part of code

**With out arguments**

In [6]:
```
n1=10
n2=20
add=n1+n2
print("the addition of {} and {} is {}".format(n1,n2,add))
```

```
the addition of 10 and 20 is 30
```

In [ ]:
```
def <function_name>():
    # write your code here
```

In [20]:
```
def addition():
    n1=10
    n2=20
    add=n1+n2
    print("the addition of {} and {} is {}".format(n1,n2,add))
```

In [21]:
```
# The output will not display untill unless you call the function
```

In [22]:
```
addition()
```

```
the addition of 10 and 20 is 30
```

In [23]:
```
def hello():
    n1=eval(input('enter number'))
    print('hello')
    print('im writing function')
```

In [24]:
```
hello()
```

```
enter number50
hello
im writing function
```

In [ ]:

In [ ]:
```
but how did python execute 2nd and not 1st function.
```

In [17]:
```
a=100
a=200

# why a is not equal to 100
```

In [18]:
```
print(a)
```

```
200
```

In [19]:
```
a=1000
```

In [ ]:
```
# python will latest value one
```

In [ ]:
```
# In this particular cell : 500lines
```

In [ ]:
```
# lin1
```

In [ ]:
```
# lin2
```

In [ ]:
```
# lin3
```

In [25]:
```
n1=100
```

In [26]:
```
n2=200
```

In [27]:
```
print(n1+n2)
```

```
300
```

In [29]:
```
def hello_world():
    print('hello good morning')

hello_world()
```

```
hello good morning
```

In [ ]:
```
# Take three numbers find the average
# WAP ask the user enter 3 numbers
# n1=eval(input())
# n2=eval(input())
# n3=eval(input())
# avg=(n1+n2+n3)/3
# First write normal code
# Then implemnt the function
```

In [30]:
```
num1=eval(input("enter number1:"))
num2=eval(input("enter number2:"))
num3=eval(input("enter number3:"))
avg=(num1+num2+num3)/3
print("the average of {},{} and {} is {}".format(num1,num2,num3,avg))
```

```
enter number1:20
enter number2:30
enter number3:40
the average of 20,30 and 40 is 30.0
```

In [37]:
```
def avg():
    num1=eval(input("enter number1:"))
    num2=eval(input("enter number2:"))
    num3=eval(input("enter number3:"))
    avg=(num1+num2+num33333)/3
    print("the average of {},{} and {} is {}".format(num1,num2,num3,avg))
```

In [38]:
```
avg()
```

```
enter number1:100
enter number2:100
enter number3:100

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[38], line 1
----> 1 avg()

Cell In[37], line 5, in avg()
      3 num2=eval(input("enter number2:"))
      4 num3=eval(input("enter number3:"))
----> 5 avg=(num1+num2+num33333)/3
      6 print("the average of {},{} and {} is {}".format(num1,num2,num3,avg))

NameError: name 'num33333' is not defined
```

**Note**:

we are not sure , the function is defined correct or wrong

untill unless we call the function

In [39]:
```
try:
    num1=eval(input("enter number1:"))
    num2=eval(input("enter number2:"))
    num3=eval(input("enter number3:"))
    avg=(num1+num2+num3)/3
    print("the average of {},{} and {} is {}".format(num1,num2,num3,avg))

except Exception as e:
    print(e)
```

```
enter number1:20
enter number2:30
enter number3:40
the average of 20,30 and 40 is 30.0
```

In [41]:
```
def avg1():
    try:
        num1=eval(input("enter number1:"))
        num2=eval(input("enter number2:"))
        num3=eval(input("enter number3:"))
        avg=(num1+num2+num3333)/3
        print("the average of {},{} and {} is {}".format(num1,num2,num3,avg))

    except Exception as e:
        print(e)

avg1()
```

```
enter number1:20
enter number2:30
enter number3:40
name 'num3333' is not defined
```

In [ ]:
```
# WAP ask the user enter bill amount
#     ask the user enter tip amount
#     calculate total bill

# First write normal code
# Then implement the function
```

In [42]:
```
def Bill():
    try:
        Bill_Amt= eval(input("Enter Bill amount"))
        Tip_Amt= eval(input("Enter Tip amount"))
        Total_Bill = Bill_Amt+Tip_Amt
        print("Total Bill is {}".format(Total_Bill))
    except exception as e:
        print(e)

Bill()
```

```
Enter Bill amount20
Enter Tip amount30
Total Bill is 50
```

In [ ]:
```
#WAP ask the user get a random number between 1 to 100
# print it is even or odd number
# Implement using function
# import <package>
# num=<package>.<method_name>(1,100)
# if <condition>:
#      print("{} is an even number".format(num))
#else:
#    print("{} is an odd number".format(num))
```

In [44]:
```
import random
num=random.randint(1,100)
if num%2==0:
    print("{} is an even number".format(num))
else:
    print("{} is an odd number".format(num))
```

```
54 is an even number
```

In [47]:
```
import random
def even_odd1():
    print('we are implementing even odd function')
    print("take one number")
    num=random.randint(1,100)
    if num%2==0:
        print("the remainder is zero")
        print("{} is an even number".format(num))
    else:
        print("the remainder is not equal to zero")
        print("{} is an odd number".format(num))

even_odd1()
```

```
wev are implementing even odd function
take one number
the remainder is zero
76 is an even number
```

In [48]:
```
import random
def even_odd2():
    try:
        print('we are implementing even odd function')
        print("take one number")
        num=random.randint(1,100)
        if num%2==0:
            print("the remainder is zero")
            print("{} is an even number".format(num))
        else:
            print("the remainder is not equal to zero")
            print("{} is an odd number".format(num))

    except Exception as e:
        print(e)

even_odd2()
```

```
we are implementing even odd function
take one number
the remainder is zero
8 is an even number
```

In [49]:
```
import random
print('hello')
print("python")
def even_odd2():
    try:
        print('we are implementing even odd function')
        print("take one number")
        num=random.randint(1,100)
        if num%2==0:
            print("the remainder is zero")
            print("{} is an even number".format(num))
        else:
            print("the remainder is not equal to zero")
            print("{} is an odd number".format(num))

    except Exception as e:
        print(e)

print('calling function')
even_odd2()
print("done!")
```

```
hello
python
calling function
we are implementing even odd function
take one number
the remainder is zero
50 is an even number
done!
```

In [ ]:
```
def addition():
    n1=20
    n2=30
    print('addition:',n1+n2)

print("mul function starts")
def mul():
    n1=20
    n2=30
    print('mul:',n1*n2)
    print("mul is done")

print("subtraction function starts")
def sub():
    n1=20
    n2=30
    print('sub:',n1-n2)
    print("subtraction is done")

print("anything remains")
print("no")
print("then call the function")

sub()
mul()
addition()
```

In [ ]:
```

```

In [ ]:
```
def addition():
    n1=10
    n2=20
    add=n1+n2
    print("the addition of {} and {} is {}".format(n1,n2,add))

addition()
```

In [ ]:
```
def Bill():
    try:
        Bill_Amt= eval(input("Enter Bill amount"))
        Tip_Amt= eval(input("Enter Tip amount"))
        Total_Bill = Bill_Amt+Tip_Amt
        print("Total Bill is {}".format(Total_Bill))
    except exception as e:
        print(e)

Bill()
```

In [ ]:
```
addition()
Bill()
```

- if you are not mentioning any values inside the function bracket
- with out arguments or with out parmeters
- arguments or parameters

**With arguments**

In [ ]:
```
def addition():
    n1=10
    n2=20
    add=n1+n2
    print("the addition of {} and {} is {}".format(n1,n2,add))

addition()

# 1Q) inside function how many variables are there
#      3 variables are there
# 2Q) how many user provided variables
#       2 variables
```

In [53]:
```
def addition(n1,n2): # arguments
    add=n1+n2
    print("the addition of {} and {} is {}".format(n1,n2,add))

addition(100,200)  # n1=100  n2=200
```

```
the addition of 100 and 200 is 300
```

In [55]:
```
def addition(n1,n2):
    add=n1+n2
    print("the addition of {} and {} is {}".format(n1,n2,add))

addition(200)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[55], line 6
      3     add=n1+n2
      4     print("the addition of {} and {} is {}".format(n1,n2,add))
----> 6 addition(200)

TypeError: addition() missing 1 required positional argument: 'n2'
```

In [57]:
```
def addition11():
    add=n111+n222
    print("the addition of {} and {} is {}".format(n1,n2,add))

addition11(200,300)   # what is the error
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[57], line 5
      2     add=n111+n222
      3     print("the addition of {} and {} is {}".format(n1,n2,add))
----> 5 addition11(200,300)

TypeError: addition11() takes 0 positional arguments but 2 were given
```

In [58]:
```
# Implement below code using with arguments
def avg(num1,num2,num3):
    print("num1:",num1) # 20
    print("num2:",num2) #30
    print("num3:",num3) # 40
    avg=(num1+num2+num3)/3
    print("the average of {},{} and {} is {}".format(num1,num2,num3,avg))

avg(20,30,40)
```

```
num1: 20
num2: 30
num3: 40
the average of 20,30 and 40 is 30.0
```

In [67]:
```
num=12345
v1=num%10     #5
v11=num//100
v11  # 1234

v2=v11%10    # 4
v2
v22=v11//10 # 123
v22

# 12345========= 1234=====4
# 54321
```

Out[67]: 12

In [69]:
```
num=12345
v1=num%100
v1
```

Out[69]: 45

In [66]:
```
v11=num//10
v11
```

Out[66]: 1234

In [64]:
```
if a=10:
    print('h')
```

```
  Cell In[64], line 1
    if a=10:
       ^
SyntaxError: invalid syntax. Maybe you meant '==' or ':=' instead of '='?
```

In [ ]:
```
7260927090
```

In [ ]:
```
12345 ======== 54
```

In [70]:
```
12345%10
```

Out[70]:
```
5
```

In [ ]:
```
12345 ======== 4?
```

In [ ]:
```
1234?
```

In [71]:
```
print(5,end='')
print(4)
```

```
54
```

In [ ]:
```
12345%10=====> 5
```

In [73]:
```
12345//10
```

Out[73]:
```
1234
```

In [78]:
```

```

Out[78]:
```
2
```

In [79]:
```
180
100
```

Out[79]:
```
100
```

In [ ]:
```
1100
150
310
```

In [ ]:
```
def Bill():
    try:
        Bill_Amt= eval(input("Enter Bill amount"))
        Tip_Amt= eval(input("Enter Tip amount"))
        Total_Bill = Bill_Amt+Tip_Amt
        print("Total Bill is {}".format(Total_Bill))
    except exception as e:
        print(e)

Bill()


# First how many important(user) variables are there inside the function
# Bill_Amy, Tip_AMT
```

In [2]:
```
def Bill(Bill_Amt,Tip_Amt):
    try:
        Total_Bill = Bill_Amt+Tip_Amt
        print("Total Bill is {}".format(Total_Bill))
    except exception as e:
        print(e)

Bill(1000,50)

# What error will come
# Name error: Bill_Amt not defined
```

```
Total Bill is 1050
```

In [ ]:
```
# WAP ask the user enter number
# and find the square of the number

# M-1: Write in normal ways
# M-2: write in function with out argument
# M-3: Weite in function with argument
```

In [3]:
```
number=eval(input("enter a number"))
print("the square of {} is {}".format(number,number*number))
```

```
enter a number5
the square of 5 is 25
```

In [4]:
```
def square():   # In side the bracket No arguments
    number=eval(input("enter a number"))
    print("the square of {} is {}".format(number,number*number))

square()
```

```
enter a number6
the square of 6 is 36
```

In [5]:
```
def square(number):
    print("the square of {} is {}".format(number,number*number))

square(7)

# How many variables are there inside the function: number
```

```
the square of 7 is 49
```

In [ ]:
```
# WAP ask there user
# get one random number between 1 to 20
# enter a number
# compare these two number
# if both are same: print("you won")
# else: print(lost)

#M-1: Write in normal way
#M-2: Create a function with out argument
#M-3: Create a function with argument
```

In [7]:
```
#M-1:
import random
random_num=random.randint(1,20)
num=eval(input("enter number:"))

if random_num==num:
    print("you won")
else:
    print("you lost")
```

```
enter number:7
you lost
```

In [9]:
```
#M-2:
import random
def compare():
    random_num=random.randint(1,20)
    num=eval(input("enter number:"))

    if random_num==num:
        print("you won")
    else:
        print("you lost because the random number is:",random_num)

compare()
```

```
enter number:9
you lost because the random number is: 17
```

In [14]:
```
#M-2:
import random
def compare(num):
    random_num=random.randint(1,20)
    if random_num==num:
        print("you won")
    else:
        print("you lost because the random number is:",random_num)

compare(eval(input("enter number"))) # keyboard pass
```

```
enter number20
you lost because the random number is: 13
```

In [13]:
```
#M-2:
import random
def compare(num):
    random_num=random.randint(1,20)
    if random_num==num:
        print("you won")
    else:
        print("you lost because the random number is:",random_num)

compare(20) # direct pass
```

```
you lost because the random number is: 19
```

In [ ]:
```
#M-2:
import random
def compare(num):
    random_num=random.randint(1,20)
    if random_num==num:
        print("you won")
    else:
        print("you lost because the random number is:",random_num)

value=20
compare(value) # direct pass
```

In [15]:
```
#M-2:
import random
def compare(random_num,num):

    if random_num==num:
        print("you won")
    else:
        print("you lost because the random number is:",random_num)

value_random=random.randint(1,20)
value=eval(input("enter number"))

compare(value_random,value)


# Step-1: import random
# step-2: function will defin: not give
# step-3: value_random
# step-4: value
# step-5: calling the function======> step-2
#                                    print won/loss
```

```
enter number7
you lost because the random number is: 18
```

In [16]:
```
# Step-1: import random
# step-3: value_random
# step-4: value
# step-2: function will defin: not give
# step-5: calling the function======> step-2
#                                    print won/loss


import random

value_random=random.randint(1,20)
value=eval(input("enter number"))

def compare(random_num,num):

    if random_num==num:
        print("you won")
    else:
        print("you lost because the random number is:",random_num)

compare(value_random,value)
```

```
enter number8
you lost because the random number is: 9
```

In [ ]:
```
# Step-1: import random
# step-2: function will defin: not give
# step-5: calling the function======> step-2
#                                    print won/loss
# step-3: value_random
# step-4: value
```

In [17]:
```
import random
random.randint(1,20)
```

Out[17]: 3

In [18]:
```
from random import randint
randint(1,20)
```

Out[18]: 16

In [25]:
```
import random

#==============================defined======================
def megha(random_num,num):

    if random_num==num:
        print("you won")
    else:
        print("you lost because the random number is:",random_num)

#================calling=============================================
#
megha(20,10)
```

```
enter number9
you lost because the random number is: 2
```

In [ ]:
```
# WAP ask the user enter salary
#     ask the user enter tax percentage
#     calulate total tax to pay
```

In [29]:
```
import random
def compare(user_num,ran_num):
    try:
        if ran_num == user_num:
            print("You won")
        else:
            print("You are not a winner")
    except exception as e:
        print(e)

#user_num=eval(input("enter number"))
#ran_num=random.randint(1,20)
#compare(user_num,ran_num)

compare(eval(input("Enter a number:")),
        random.randint(1,20))
```

```
Enter a number:9
You are not a winner
```

In [31]:
```
def tax_cal(salary,tax_per):
    total_tax=salary*tax_per/100
    print("The total tax is:",total_tax)

tax_cal(10000,10)
```

```
The total tax is: 1000.0
```

In [ ]:
```
# How do you pass the value
def tax_cal(salary,tax_per):
    total_tax=salary*tax_per/100
    print("The total tax is:",total_tax)

tax_cal(10000,10)                # Direct pass
```

In [33]:
```
# How do you pass the value
def tax_cal(salary,tax_per):
    total_tax=salary*tax_per/100
    print("The total tax is:",total_tax)

val1=eval(input("enter salary:"))
val2=eval(input("enter tax perc:"))
tax_cal(val1,val2)                # keyboard pass
```

```
enter salary:10000
enter tax perc:10
The total tax is: 1000.0
```

**default arguments**

In [ ]:
```
# tax percentage is always=20
# fixed
# default parameter
```

In [34]:
```
# How do you pass the value
def tax_cal(salary,tax_per=20):
    total_tax=salary*tax_per/100
    print("The total tax is:",total_tax)

tax_cal(10000)
```

```
The total tax is: 2000.0
```

In [36]:
```
def avg(num1,num2,num3):
    add=(num1+num2+num3)
    avg=add/3
    print("the avg is:",avg)

avg(20,30,40)
```

```
the avg is: 30.0
```

In [43]:
```
def avg(num1,num2,num3=50):
    print("num1:",num1)           # 20
    print("num2:",num2)           # 20
    print("num3:",num3)            # 100
    add=(num1+num2+num3)
    avg=add/3
    print("the avg is:",avg)

avg(20,20,100)
```

```
num1: 20
num2: 20
num3: 100
the avg is: 46.666666666666664
```

In [45]:
```
def avg(num1,num2=200,num3):
    print("num1:",num1)           #
    print("num2:",num2)           #
    print("num3:",num3)            #
    add=(num1+num2+num3)
    avg=add/3
    print("the avg is:",avg)

avg(200,300)
```

```
  Cell In[45], line 1
    def avg(num1,num2=200,num3):
                              ^
SyntaxError: non-default argument follows default argument
```

In [46]:
```
def avg(num1,num2,num3=50):
    print("num1:",num1)           # 20
    print("num2:",num2)           # 20
    print("num3:",num3)            # 100
    add=(num1+num2+num3)
    avg=add/3
    print("the avg is:",avg)

va1=20
val2=30
avg(val1,val2)
```

```
num1: 10000
num2: 30
num3: 50
the avg is: 3360.0
```

In [ ]:

In [ ]:

In [ ]:

In [42]:
```
num1,num2,num3=100
print(num1,num2,num3)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[42], line 1
----> 1 num1,num2,num3=100
      2 print(num1,num2,num3)

TypeError: cannot unpack non-iterable int object
```

In [ ]:
```
num1,num2,num3=300  =========== valid
num1,num2=200,num3  ========== not
num1=100,num2=200,num3 ====== not
num1=100,num2,num3 ========= not
num1,num2=500,num3=300 ===== valid
```

In [ ]:
```
def avg(num1,num2,num3=50):
    print("num1:",num1)           # 20
    print("num2:",num2)           # 20
    print("num3:",num3)            # 100
    num3=800
    add=(num1+num2+num3)
    avg=add/3
    print("the avg is:",avg)

avg(200,300,600)

# while defining function: num3=50
# while you are calling function num3=600
# after enter inside the function num3=800
```

- with out arguments
- with arguments
- default arguments

In [50]:
```
import random
random.randint()   # a

# (<excpecting 2 arguments>)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[50], line 2
      1 import random
----> 2 random.randint()

TypeError: Random.randint() missing 2 required positional arguments: 'a' and 'b'
```

In [54]:
```
complex(7) # not provide: default
```

Out[54]:
```
(7+0j)
```

In [56]:
```
from random import randint
randint(10,20)
```

Out[56]:
```
19
```

In [ ]:

In [ ]:
```
# function = method
```

In [58]:
```
def summ(a=0,b=0):
    print(a*b)

summ(5,10)
```

```
50
```

In [ ]:
```
# WAP ask the user to find area of circle
# Formulae: pi*radius*radius
# import math, math.pi
# Basic method
# With out arguments
# With arguments
# Default arguments
```

In [5]:
```
import math
pii=math.pi
r=eval(input("enter the radius"))
area1=round(pii*r*r,2)
print("the area of circle is:",area1)
#area2=pii*r**2
```

```
enter the radius20
the area of circle is: 1256.64
```

In [7]:
```
# M-1: with out arguments
import math
def area_of_circle():
    pii=math.pi
    r=eval(input("enter the radius"))
    area1=round(pii*r*r,2)
    print("the area of circle is:",area1)

area_of_circle()
```

```
enter the radius20
the area of circle is: 1256.64
```

In [8]:
```
# M-2: with arguments  direct pass
import math
def area_of_circle(r):
    pii=math.pi
    area1=round(pii*r*r,2)
    print("the area of circle is:",area1)

area_of_circle(20) # Direct pass
```

```
the area of circle is: 1256.64
```

In [9]:
```
# M-2: keyboard pass
# you need to provide argument value before call the function anywhere
#r=eval(input("enter radius"))
import math                          # s-1
def area_of_circle(r):
    pii=math.pi
    area1=round(pii*r*r,2)
    print("the area of circle is:",area1)
r=eval(input("enter radius"))
area_of_circle(r)

# s-1: import math
# s-2: define function
# s-3: getting r value
# s-4: call the function
# s-5:            pii value
# s-6:            area1
#s-7:             print
```

```
enter radius20
the area of circle is: 1256.64
```

In [10]:
```
# M-2: keyboard pass
# you need to provide argument value before call the function anywhere
#r=eval(input("enter radius"))
import math                          # s-1
def area_of_circle(r):
    pii=math.pi
    area1=round(pii*r*r,2)
    print("the area of circle is:",area1)

area_of_circle(eval(input("enter radius")))

# s-1: import math
# s-2: define function
# s-3: getting r value
# s-4: call the function
# s-5:            pii value
# s-6:            area1
#s-7:             print
```

```
enter radius20
the area of circle is: 1256.64
```

In [11]:
```
# M-3: deafualt arguments
import math
def area_of_circle(pii,r=20):
    area1=round(pii*r*r,2)
    print("the area of circle is:",area1)

pii=math.pi
area_of_circle(pii)   # 3.14=pii
```

```
the area of circle is: 1256.0
```

In [14]:
```
import math
def area_of_circle():
    try:
        pii=math.pi
        r=eval(input("enter the radius"))
        area11=round(pii*r*r,2)
        print("the area of circle is:",area11)

    except Exception as e:
        print(e)

area_of_circle()
```

```
enter the radius20
the area of circle is: 1256.64
```

In [15]:
```
area11

# Function is calculating area value
# but not providing(return) that value to you
# when you want use that value outside the function========> error
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[15], line 1
----> 1 area11

NameError: name 'area11' is not defined
```

**Return**

In [20]:
```
import math
def area_of_circle():
    pii=math.pi
    r=eval(input("enter the radius"))
    area11=round(pii*r*r,2)
    print("the area of circle is:",area11)
    return(area11,pii)
```

In [21]:
```
Area,Pi=area_of_circle()  #

# area_of_circle() will return two values
# area11,pii
# area11 will store in a variable: Area
# pii will store in a variable: Pi
```

```
enter the radius20
the area of circle is: 1256.64
```

In [22]:
```
Area
```

Out[22]:
```
1256.64
```

In [23]:
```
Pi
```

Out[23]:
```
3.141592653589793
```

In [24]:
```
import math
def area_of_circle():
    pii=math.pi
    r=eval(input("enter the radius"))
    area11=round(pii*r*r,2)
    print("the area of circle is:",area11)
    return(area11,pii)

Area,Pi=area_of_circle()
print(Area)
print(Pi)
```

```
enter the radius20
the area of circle is: 1256.64
1256.64
3.141592653589793
```

In [ ]:
```
# ask the user enter 3 numbers
# find the  sum and average
# Return the avergae value and sum both
```

In [27]:
```
def sum_avg(n1,n2,n3):
    summ=n1+n2+n3
    avg=summ/3
    print('sum:',summ)
    print('avg:',avg)
    return(summ,avg)

addition,average=sum_avg(20,30,40)

# sum_avg(20,30,40) is returning two values
# summ and avg
# summ we are saving in a variable: addition
# avg we are saving in a variable: average
print('addition is:',addition,'average is:',average)
```

```
sum: 90
avg: 30.0
addition is: 90 average is: 30.0
```

In [ ]:
```
# Find the biggest number of 3 numbers
# take three numbers
# and find the biggest number
# and return that number

#if num1>num2 and num1>num3 : num1

#elif num2>num3:  num2

# esle:  num3

# first write the normal code
# then create the function
# return the value
```

In [30]:
```
def greater(n1,n2,n3):
    if n1>n2 and n1>n3:
        print('{} is greater'.format(n1))
        return(n1)

    elif n2>n3:
        print("{} is greater".format(n2))
        return(n2)
    else:
        print("{} is greater".format(n3))
        return(n3)

GREATER=greater(20,200,10)

# It is returning only one values
# whenever condition satisfi

print(GREATER)
```

```
200 is greater
200
```

In [ ]:
```
def numbers():
    num1=eval(input("Enter 1st no"))
    num2=eval(input("Enter 2nd no"))
    num3=eval(input("Enter 3rd no"))
    if num2<num1>num3:
        print(num1, "is the grestest number")
        a=num1
    elif num2>num3:
        print(num2, "is the grestest number")
        a=num2
    else:
        print(num3, "is the grestest number")
        a=num3
    return(a)
a=numbers()
print(a)….got my error this is working now
```

In [ ]:
```
def sum_avg(n1,n2,n3):
    return(n1+n2+n3,(n1+n2+n3)/3)

addition,average=sum_avg(20,30,40)


def sum_avg(n1,n2,n3):
    summ=n1+n2+n3
    avg=summ/3
    print('sum:',summ)
    print('avg:',avg)
    return(summ,avg)

addition,average=sum_avg(20,30,40)
```

In [33]:
```
def add(a,b):
    return(a+b)

def mul(a,b):
    return(a*b)

def sub(a,b):
    return(a-b)

def div(a,b):
    return(a/b)

val_add=add(20,30)
val_sub=sub(20,30)
val_mul=mul(20,30)
val_div=div(20,30)
print(val_add,val_sub,val_mul,round(val_div,2))
```

```
50 -10 600 0.67
```

In [34]:
```
def aggregation(a,b):
    return(a+b,a-b,a*b,a/b)

add,sub,mul,div=aggregation(20,30)
print(add,sub,mul,round(div,2))
```

```
50 -10 600 0.67
```

In [35]:
```
def aggregation(a,b):
    return(a+b,a-b,a*b,a/b)

value=aggregation(20,30)
value
```

Out[35]:
```
(50, -10, 600, 0.6666666666666666)
```

In [ ]:
```
how to print in next line using one print statement
```

In [38]:
```
def add(a,b):return(a+b)
def sub(a,b):return(a-b)
def mul(a,b):return(a*b)
def div(a,b):return(a/b)
val_add=add(20,30)
val_sub=sub(20,30)
val_mul=mul(20,30)
val_div=div(20,30)
print(val_add,val_sub,val_mul,round(val_div,2))
```

```
50 -10 600 0.67
```

- with out arguments
- with arguments
- Defaualt arguments
- Return statements

**Local Variables**

In [1]:
```
def addition():
    n1=10
    n2=20
    add=n1+n2
    print("the addition of {} and {} is {}".format(n1,n2,add))
```

**local variables :**

```
    - The variables provided inside the function
```

**global varaibes :**

```
    The variables provided outside the function
```

In [2]:
```
n1=20
n2=30
def addition():      # No need to provide inside brackets
    # No need to mention
    add=n1+n2
    print("the addition of {} and {} is {}".format(n1,n2,add))

addition()
```

```
the addition of 20 and 30 is 50
```

In [8]:
```
# Implement below code using with arguments
def avg():
    num11=20
    num22=30
    num33=40
    avg=(num1+num2+num3)/3
    print("the average of {},{} and {} is {}".format(num1,num2,num3,avg))
    return(num11,num22)

val1,val2=avg()
```

```
the average of 20,30 and 40 is 30.0
```

In [9]:
```
val2
```

Out[9]:
```
30
```

- local variables are inside the function
- local variable values can not use outside the function

- untill unless you return those values

In [ ]:

In [ ]:

In [ ]:

In [12]:
```
num111=20
num222=30
num333=40
def avg():
    avg=(num111+num222+num333)/3
    print("the average of {},{} and {} is {}".format(num111,num222,num333,avg))

avg()

print(num111*num222)
```

```
the average of 20,30 and 40 is 30.0
600
```

In [11]:
```
num111
```

Out[11]:
```
20
```

In [13]:
```
n1=20
n2=30
def addition():
    n1=200
    n2=300
    add=n1+n2
    print("the addition of {} and {} is {}".format(n1,n2,add))

addition()

print(n1)
```

```
the addition of 200 and 300 is 500
20
```

In [14]:
```
n1=20
n2=30
def addition(n1):
    n2=300
    add=n1+n2
    print("the addition of {} and {} is {}".format(n1,n2,add))

addition(500)  # 800

print(n1)     # 20
```

```
the addition of 500 and 300 is 800
20
```

In [15]:
```
n1=20
n2=30
def addition(n1=700):
    n2=300
    add=n1+n2
    print("the addition of {} and {} is {}".format(n1,n2,add))

addition(500)

print(n1)

# step-1:  n1=20
# step-2: n2=30
# step-3: function defined n1:700
# step-4: calling the function:n1=500 =======>
# step-5:                      n2=300
#step-6:                       800
#step-7:n1=20
```

```
the addition of 500 and 300 is 800
20
```

In [ ]:
```
def numbers():
    num1=eval(input("Enter 1st no"))
    num2=eval(input("Enter 2nd no"))
    num3=eval(input("Enter 3rd no"))
    if num2<num1>num3:
        print(num1, "is the grestest number")
        a=num1
    elif num2>num3:
        print(num2, "is the grestest number")
        a=num2
    else:
        print(num3, "is the grestest number")
        a=num3
    return(a)
a=numbers()


Q1) when the concept local golbal
```

In [ ]:
```
num1=eval(input("Enter 1st no"))
num2=eval(input("Enter 2nd no"))
num3=eval(input("Enter 3rd no"))
if num2<num1>num3:
    print(num1, "is the grestest number")
    a=num1
elif num2>num3:
    print(num2, "is the grestest number")
    a=num2
else:
    print(num3, "is the grestest number")
    a=num3
return(a)
```

In [ ]:
```
# we can not use local variable values outside the function
# untill unless with out return statement

# i want to use local variable outside the function, with out using return statement
```

In [ ]:
```
global
```

In [ ]:
```
n1=10
def upadted():
    n1=n1+100
    print("n1 value inside function is:",n1)

updated()  # 110
print(n1) # 10

# with out return n1 , we need to get the updated value
# we need to use local variable outside the function, with out retutn
```

In [20]:
```
def value():
    global number1,number2
    number1=10
    number2=20
    print(number1)

value()
print(number1)
print(number2)
```

```
10
10
20
```

In [27]:
```
n_=10
def updated():
    global n_
    n_=20
    n_=n_+100
    print("n1 value inside function is:",n_)

updated()  # 110
print(n_) # 120
```

```
n1 value inside function is: 120
120
```

In [29]:
```
n1=20
n2=30
def addition(n1):
    global n2,add
    n2=300
    add=n1+n2
    print("the addition of {} and {} is {}".format(n1,n2,add))

addition(500)

print(n2)  # i need to get n2 value 300, instead of 30
print(add) # i need to add value
```

```
the addition of 500 and 300 is 800
300
800
```

**Function in function**

In [32]:
```
def greet():
    print("hello")

def name():
    print('python')
    print('how do you do')

greet()
name()
```

```
hello
python
how do you do
```

In [31]:
```
def greet():
    print("hello")

def name():
    print('python')
    greet()


name()

# step-1: define greet()
# step-2: define name()
# step-3: calling name()
# step-4: python
# step-5: calling greet()
# step-6: hello
```

```
python
hello
```

In [33]:
```
def greet():
    print("hello")
    name()

def name():
    print('python')


greet()
```

```
hello
python
```

In [35]:
```
def greet():
    print("hello")
    name()

def name():
    print("python")
    greet()

greet()  # infinite loop
print('---------')
name()    # name() this will not execute
```

```
Cell In[35], line 6, in name()
      5 def name():
----> 6     print("python")
      7     greet()

File ~\anaconda3\Lib\site-packages\ipykernel\iostream.py:635, in OutStream.write(self, string)
    633     raise ValueError(msg)
    634 else:
--> 635     is_child = not self._is_master_process()
    636     # only touch the buffer in the IO thread to avoid races
    637     with self._buffer_lock:

File ~\anaconda3\Lib\site-packages\ipykernel\iostream.py:506, in OutStream._is_master_process
(self)
    505 def _is_master_process(self):
--> 506     return os.getpid() == self._master_pid

RecursionError: maximum recursion depth exceeded while calling a Python object
```

In [ ]:

In [ ]:

```
################################################################################
def greet():
    print("hello")

def name():
    print('python')
    print('how do you do')

greet()
name()

#########################################################################################
def greet():
    print("hello")

def name():
    print('python')
    greet()


name()

#################################################################################################

def greet():
    print("hello")
    name()

def name():
    print('python')


greet()

#########################################################################################################
def greet():
    print("hello")
    name()

def name():
    print("python")
    greet()

greet()  # infinite loop
print('---------')
name()    # name() this will not execute
```

In [36]:
```
print(3)

def fun1():
    print('hello')
    print('python')

print('10+10')
def fun2():
    print('3+5')
    print(3+5)
    fun1()
    print('good')

print(10+10)
def fun3():
    fun1()
    fun2()

fun3()

# 3
# '10+10'
# 20
# 'hello'
# 'python'
# '3+5'
# 8
# 'hello'
# 'python'
# 'good'
```

```
3
10+10
20
hello
python
3+5
8
hello
python
good
```

In [ ]:
```
str1='10'
str2='20'
```

In [37]:
```
str1+str2
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[37], line 1
----> 1 str1+str2

NameError: name 'str1' is not defined
```

In [ ]:
```
# Create four functions
# def add(a+b):
#     print(a+b)
#  sub
#  mul
#  div

# def main():
    # you need to provide some print statements
    # if you want enter 1 :addition operation willcome
    # if you want enter 2: subtract
    # if you enter 3: mul
    # if you enter 4: div
    # operaion=input("enter a number")
    # a=eval(input())
    # b=eval(inpu())
    # if operation=='1':
    #      add(a,b)
    # if operation=='2':
    #      sub(a,b)
    # if operation=='3':
    #      mul(a,b)
    # if operation=='4':
    #      div(a,b)
```

In [42]:
```
a=eval(input("enter number1: "))
b=eval(input("enter a number2: "))
def add():
    print(a+b)
def sub():
    print(a-b)
def div():
    print(a/b)
def mul():
    print(a*b)
def main():
    num=eval(input("enter a number: "))

    if num==1:
        add()
    elif num==2:
        sub()
    elif num==3:
        div()
    else:
        mul()

main()
```

```
enter number1: 1
enter a number2: 20
enter a number: 30
20
```

In [ ]:
```
def add(a,b):
    print(a+b)
def sub(a,b):
    print(a-b)
def div(a,b):
    print(a/b)
def mul(a,b):
    print(a*b)
def main():
    num=eval(input("enter a number: "))
    a=eval(input("enter number1: "))
    b=eval(input("enter a number2: "))
    if num==1:
        add(a,b)
    elif num==2:
        sub(a,b)
    elif num==3:
        div(a,b)
    else:
        mul()

main()
```

In [ ]:
```
share your screenshot into my whatsapp number
```

In [ ]:
```


def add(a,b):
    print(a+b)
def sub(a,b):
    print(a-b)
def mul(a,b):
    print(a*b)
def div(a,b):
    print(a/b)

def main():
    print("For addition enter 1")
    print("For subtraction enter 2")
    print("For multiplication enter 3")
    print("For division enter 4")
    op=eval(input("Enter the operation you want to perform: "))
    a=eval(input("enter 1st number: "))
    b=eval(input("enter 2nd number: "))
    if op==1:
        add(a,b)
    if op==2:
        sub(a,b)
    if op==3:
        mul(a,b)
    if op==4:
        div(a,b)

main()
```