In [1]:
```
def add_number():
    number1 = int(input("Enter the 1st number"))
                                                    #  this a function ,Define Fu
nction
    number2 = int(input("Enter the 2nd number"))

    print(number1 + number2)
```

In [2]:
```
add_number()          #     Function call
```

```
Enter the 1st number45
Enter the 2nd number55
100
```

In [3]:
```
add_number()
```

```
Enter the 1st number20
Enter the 2nd number30
50
```

In [4]:
```
add_number()
```

```
Enter the 1st number100
Enter the 2nd number200
300
```

In [5]:
```
def add_number():            #   these values are hard coded, fixed values in
 function
    print(200+300)
```

In [6]:
```
add_number()
```

```
500
```

In [7]:
```
def sub_number():
    number1 = int(input("Enter the 1st number"))

    number2 = int(input("Enter the 2nd number"))

    number5 = number1 - number2

    print(number5)
```

In [8]:
```
sub_number()
```

```
Enter the 1st number200
Enter the 2nd number120
80
```

In [9]:
```
add_number()
```

```
500
```

In [12]:
```
sub_number()
```

```
Enter the 1st number200
Enter the 2nd number190
10
```

In [20]:
```
def add_numbers():
    number1 =int(input("Enter 1st number"))

    number2 = int(input("Enter 2nd number"))

    total = number1 + number2

    print(total)
```

In [18]:
```
add_number()
```

```
500
```

In [21]:
```
add_numbers()
```

```
Enter 1st number12
Enter 2nd number45
57
```

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

```
Enter 1st number200
Enter 2nd number700
900
```

In [30]:
```
def mul():

    num = 3*6

    print(num)
```

In [31]:
```
mul()
```

```
18
```

In [32]:
```
mul()
```

```
18
```

In [33]:
```
def multiple():
    num1 = int(input("Enter the 1st number"))

    num2 = int (input("Enter the 2nd number"))

    total = (num1 * num2)

    print(total)
```

In [34]:
```
multiple()
```

```
Enter the 1st number4
Enter the 2nd number6
24
```

In [35]:
```
multiple()
```

```
Enter the 1st number25
Enter the 2nd number50
1250
```

In [36]:
```
def multiple():
    num1 = float(input("Enter the 1st value"))

    num2 = int(input("Enter the 2nd number"))

    total = (num1 * num2)

    print(total)
```

In [39]:
```
multiple()
```

```
Enter the 1st value23.9
Enter the 2nd number23
549.6999999999999
```

In [40]:
```
multiple()
```

```
Enter the 1st value99.9
Enter the 2nd number99
9890.1
```

In [41]:
```
multiple()                    #  given both value are integer but result in flo
at
                              #  because num1 in function i declare in floatin
 value
```

```
Enter the 1st value13
Enter the 2nd number13
169.0
```

In [44]:
```
def div():
    a = 90

    b = 15

    result = a/b          #  result in floating number by defult in python

    print(result)          #   result = a//b  , the result should be in intege
r
```

In [43]:
```
div()
```

```
6.0
```

In [46]:
```
div()
```

```
6
```

In [47]:
```
def divide():
    a = 990

    b = 20

    result = a//b

    print(result)
```

In [48]:
```
divide()        #  So the result in integer
```

```
49
```

In [49]:
```
divide()
```

```
49
```

In [50]:
```
def reminder():
    num1 = int(input("Enter the first value"))

    num2 = int(input("Enter the 2nd value"))

    total = num1 % num2

    print(total)
```

In [51]:
```
reminder()
```

```
Enter the first value200
Enter the 2nd value10
0
```

In [52]:
```
reminder()
```

```
Enter the first value1000
Enter the 2nd value25
0
```

In [53]:
```
reminder()
```

```
Enter the first value444
Enter the 2nd value13
2
```

In [54]:
```
reminder()
```

```
Enter the first value76451
Enter the 2nd value83
8
```

PASSING INFORMATION BY POSITIONAL ARGUMENTS

In [55]:
```
#   there are two type of functions
#1     parameter less function
#2     parameterized function
```

Parameter less Function

In [56]:
```
#     def       is a key-word
#     function name
#      bracket  or parenthysis
#     colon
```

In [58]:
```
#   def add_number():    #  if parenthysis is empty ,its mean that
                      #   it is a parameter less function
```

PARAMETERIZED FUNCTION

In [59]:
```
def add_numbers(num1, num2):
                              #     variables name or value in parenthysis are c
known as PARAMETERS
    print(num1+num2)
```

In [61]:
```
add_numbers(3, 9)
```

```
12
```

In [62]:
```
add_numbers(83,91)            #  Function call ,in parenthesis value are know
ns as ARGUMENTS
```

```
174
```

In [63]:
```
def sub(num1, num2):

    print(num1-num2)
```

In [64]:
```
sub(200,100)
```

```
100
```

In [65]:
```
sub(200,300)              # these are positional arguments
                          #   num1 = 200
                         #    num2  = 300
```

```
-100
```

In [66]:
```
sub(200,23.45)
```

```
176.55
```

In [67]:
```
def mul(a,b):
    print(a*b)
```

In [68]:
```
mul(3,7)
```

```
21
```

In [69]:
```
mul(100,5)
```

```
500
```

In [70]:
```
def divide(a,b):

    print(a/b)
```

In [71]:
```
divide(100,20)
```

```
5.0
```

In [72]:
```
divide(200,12)
```

```
16.666666666666668
```

In [73]:
```
def divide(a,b):

    print(a//b)
```

In [74]:
```
divide(200,20)
```

```
10
```

In [75]:
```
divide(290,19)
```

```
15
```

KEY-WORD ARGUMENTS

In [76]:
```
def add_numbers(number1,number2):

    print(number1+number2)             # So these are key-words arguments
```

In [77]:
```
add_numbers(number2=200, number1=100)     # in key-words arguments position no
t matter or should be changed
```

```
300
```

In [ ]:
```

```

In [91]:
```
def full_name(first,middle,last):

    print(first+middle+last)
```

In [92]:
```
full_name("RANA","SAEED","UTTERA")
```

```
RANASAEEDUTTERA
```

In [93]:
```
full_name("MR","RANA","RAMZAN")   # these are positional arguments
```

```
MRRANARAMZAN
```

In [94]:
```
#     KEY-WORD ARGUMENTS
```

In [95]:
```
def fullName(middle="Muhammad",last="Saeed",first="Rana"):

    print(first+middle+last)
```

In [96]:
```
fullName()
```

```
RanaMuhammadSaeed
```

In [97]:
```
fullName()
```

```
RanaMuhammadSaeed
```

In [98]:
```
def fullName(first,middle,last="Saeed"):

    print(first+middle+last)
```

In [99]:
```
fullName("MR","Rana")
```

```
MRRanaSaeed
```

DEFAULT PARAMETERS

In [100]:
```
def fullName(first,middle,last):

    print(first+middle+last)
```

In [102]:
```
fullName("Rana","Saeed")    #  it should be generate error because rana is fir
st and middle is saeed and last is default
                            #   operators
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-102-09c900b19518> in <module>
----> 1 fullName("Rana","Saeed")    #  it should be generate error because ra
na is first and middle is saeed and last is default
      2                             #   operators

TypeError: fullName() missing 1 required positional argument: 'last'
```

In [105]:
```
def fullName(first,last,middle=" "):

    print(first+middle+last)
```

In [106]:
```
fullName("Rana","Saeed")
```

```
Rana Saeed
```

In [107]:
```
def fullName(last,middle,first=" "):

    print(first+middle+last)
```

In [108]:
```
fullName("Rana","Saeed")
```

```
 SaeedRana
```

DEALING WITH AN UNKNOWN NUMBER OR ARBITARARY NUMBER

In [2]:
```
def pizzaorder(size,flavour,toppings):

    print(f"your order for pizza of{size},flavour{flavour},and toppings{toppin
gs} is ready")
```

In [3]:
```
pizzaorder(12,"chikentikka","olives","fruits","tea")   # we do not know that t
he user gives more values
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-3-c15a80f448b8> in <module>
----> 1 pizzaorder(12,"chikentikka","olives","fruits","tea")

TypeError: pizzaorder() takes 3 positional arguments but 5 were given
```

In [6]:
```
def pizzaorder(size,flavour,*toppings):           #   we first define topping
s with steric

    print(f"your order for pizza of{size},flavour{flavour},and toppings{toppin
gs} is ready")
```

In [7]:
```
pizzaorder(12,"chikentikka","olives","fruits","tea")
```

```
your order for pizza of12,flavourchikentikka,and toppings('olives', 'fruits',
'tea') is ready
```

In [8]:
```
#      here we can see that more than three value are print ,so we define ster
ic with toppings
```

PASSING INFORMATION BACK FROM THE FUNCTION

In [9]:
```
def add_numbers(num1,num2):

    ans = num1+num2

    return  ans
```

In [13]:
```
result = add_numbers(23,27)

result
```

Out[13]: 50

In [14]:
```
result *10         #  we can perform other task because we store value in vari
ables in result
```

Out[14]: 500

In [17]:
```
def add_numbers(num1,num2):

    ans = num1+num2

    return  ans, "RANA PLEASE KEEP TRYING PRACTICE IF WANT SUCCESS"
```

In [18]:
```
result = add_numbers(70,20)
                                #     RESULT IN THE FORM OF TUPLE IF VALUES ARE
MORE
result
```

Out[18]:
```
(90, 'RANA PLEASE KEEP TRYING PRACTICE IF WANT SUCCESS')
```

In [19]:
```
result = add_numbers(100,400)
```

In [20]:
```
result
```

Out[20]:
```
(500, 'RANA PLEASE KEEP TRYING PRACTICE IF WANT SUCCESS')
```

In [24]:
```
result
```

Out[24]:
```
(500, 'RANA PLEASE KEEP TRYING PRACTICE IF WANT SUCCESS')
```

In [25]:
```
result *2    # in this case value not multiply because result in tuple ,so tup
le not multiply but they print 2 times the result
```

Out[25]:
```
(500,
 'RANA PLEASE KEEP TRYING PRACTICE IF WANT SUCCESS',
 500,
 'RANA PLEASE KEEP TRYING PRACTICE IF WANT SUCCESS')
```

In [27]:
```
result = add_numbers(29,31)

result *2
```

Out[27]:
```
(60,
 'RANA PLEASE KEEP TRYING PRACTICE IF WANT SUCCESS',
 60,
 'RANA PLEASE KEEP TRYING PRACTICE IF WANT SUCCESS')
```

In [28]:
```
def add_numbers(num1,num2):

    ans = num1+num2

    return  ans
```

In [29]:
```
result = add_numbers(21,29)
result*2
```

Out[29]:
```
100
```

In [30]:
```
def add_numbers(num1,num2):

    ans = num1+num2

    return  ans
```

In [31]:
```
result = add_numbers(29,21)
                #            we can add,multiply ,subtract and divide vale fro
m return function
result+50
```

Out[31]: 100

In [33]:
```
result = add_numbers(29,21)
result-30
```

Out[33]: 20

In [36]:
```
result = add_numbers(30,90)    # the result in floats
result/20
```

Out[36]: 6.0

In [37]:
```
result = add_numbers(30,90)  # the result in integer
result//20
```

Out[37]: 6

In [38]:
```
result = add_numbers(90,30)
result%25
```

Out[38]: 20

In [39]:
```
result = add_numbers(120,80)
result%15                    #  so we can perform some tasks
```

Out[39]: 5

USING FUNCTION AS VARIABLES

In [41]:
```
def add_numbers(a,b):
    return a+b


def sub_numbers(b,a):
    return a-b
```

In [42]:
```
result = add_numbers(15,5) + sub_numbers(15,30)
                             #    here these two functions are used as variab
les
```

In [43]:
```
result
```

Out[43]: 35

In [44]:
```
def multi_numbers(a,b):
    return a*b


def divide_numbers(a,b):
    return a/b
```

In [45]:
```
result = multi_numbers(12,13) % divide_numbers(20,5)
result
```

Out[45]: 0.0

In [49]:
```
result = multi_numbers(15,14) % divide_numbers(20,5)
                              #   we can use modulos operator used also
result
```

Out[49]: 2.0

LOCAL AND GLOBAL VARIABLES

In [56]:
```
def beHappy():
    name =" Mr A"
    #            name variable define inside the function ,so it is a local va
riable

    print(f"{name} is very happy today")
```

In [57]:
```
beHappy()
```

```
 Mr A is very happy today
```

In [58]:
```
print(name)    #   error occurs,name variable not define because its scope is
 inside the function
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-58-9ba126b17b03> in <module>
----> 1 print(name)

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

In [59]:
```
another_name = "Mr B"
            #         another_name variable have declear outside the function
 ,so it have scope outside the fuction
def sad():

    print(f"{another_name} is very sad today")
```

In [60]:
```
sad()
```

```
Mr B is very sad today
```

In [61]:
```
print(another_name)  #   it means that another_name variable is Global variables
```

```
Mr B
```

FUNCTION WITHIN FUNCTION

In [78]:
```
    def add_number():
        print(90+80)
```

In [79]:
```
add_number()
```

```
170
```

In [80]:
```
    add_number()
```

```
170
```

In [81]:
```
def add():
    print(20+12)
```

In [82]:
```
add()
```

```
32
```

In [92]:
```
def commissioncalc(sales):
    if sales>100 :
        return sales*100
    elif sales>50:
        return sales*50
    elif sales>20:
        return sales*20
    else:
        return 0

    def salarycalc(basics,sales):

        grosssalary = basics + commissioncalc(sales)

        print(f"your gross salary is {grosssalary}")
```

In [93]:
```
salarycalc(20000,120)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-93-50fa904a6bcd> in <module>
----> 1 salarycalc(20000,120)

<ipython-input-74-7afc0ddaa92a> in salarycalc(basics, sales)
     15     grosss_salary = basics + commissioncalc(sales)
     16
---> 17     print(f"your gross salary is {gross_salary}")
     18
     19

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

In [94]:
```
salarycalc(20000,120)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-94-50fa904a6bcd> in <module>
----> 1 salarycalc(20000,120)

<ipython-input-74-7afc0ddaa92a> in salarycalc(basics, sales)
     15     grosss_salary = basics + commissioncalc(sales)
     16
---> 17     print(f"your gross salary is {gross_salary}")
     18
     19

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

In [ ]:
```

```