Functions

- Reuse of the code
- imagine you want to calculate tax of all tax payers.suppose there are 100 taxes payers
- in order to 1 tax calculate we require 5 lines
- for 100 tax payers it requires 500 lines
- this is not the good approach
- because 5 lines we are repeating for 100 members
- Here Functions will help to use only these 5 lines to all 100 tax payers.

In [1]:
```
salary=eval(input("Enter the salary"))
tax_per=eval(input("Enter the tax  per:"))
tax_amount=salary*tax_per/100
print(f" The amount of tax pay is {tax_amount}")
```

```
The amount of tax pay is 5000.0
```

In [ ]:
```
##syntax##

# def <function_name>():
#      <code stats here>
```

In [1]:
```
n1=10
n2=20
add=n1+n2
print(f"{add}")
```

```
30
```

In [4]:
```
def addition():
    n1=10
    n2=20
    add=n1+n2
    print(f"{add}")
```

In [5]:
```
addition()
```

```
30
```

In [6]:
```
def sruthi():
    n1=10
    n2=20
    add=n1+n2
    print(f"{add}")
```

In [7]:
```
sruthi()
```

```
30
```

common mistakes

- forgetting the brackets while defining the functions
- brackets() and colmn: will miss

- indentation mistake
- allignment mistake i.e copy the next lines and press tab
- your variable name and variable should be different
- while calling the function we will forget the brackets()
- function also called as a method
- whenever function is there brackets() must

In [9]:
```
##error trail##
def addition:
    n1=10
    n2=20
    add=n1+n2
    print(f"{add}")
```

```
  Cell In[9], line 1
    def addition:
                 ^
SyntaxError: expected '('
```

In [10]:
```
#error trail#
def addition()
    n1=10
    n2=20
    add=n1+n2
    print(f"{add}")
```

```
  Cell In[10], line 1
    def addition()
                  ^
SyntaxError: expected ':'
```

In [11]:
```
def addition():
    n1=10
n2=20
add=n1+n2
print(f"{add}")
```

```
30
```

In [13]:
```
#error trail#
def addition ():
n1=10
    n2=20
    add=n1+n2
    print(f"{add}")
```

```
  Cell In[13], line 2
    n1=10
    ^
IndentationError: expected an indented block after function definition on line 1
```

In [14]:
```
def addition():
    n1=10
    n2=20
```

```
    add=n1+n2
    print(f"{add}")
```

In [15]:
```
addition
```

Out[15]:
```
<function __main__.addition()>
```

In [20]:
```
def addition():
    n1=10
n2=20
add=n1+n2
print(f"{add}")
```

```
30
```

Note

- when we define it will not give any error
- after calling it gives error

METHOD-1

functions without arguments

- when we provide brackets will not provide any values ( )
- these values are called"arguments" or "parameters"
- if we don't provide arguments ======called as "function without arguments""

In [3]:
```
# create a function read three numbers and find the avg##

def average():
    n1=eval(input("enter a number1:"))
    n2=eval(input("enter a number2:"))
    n3=eval(input("enter a number3:"))
    avg=(n1+n2+n3)/3
    print(f"the given numbers are {n1},{n2},{n3} and average is{avg}")
average()
```

```
the given numbers are 2,4,6 and average is4.0
```

In [5]:
```
## create a function ask the user enter the bill and tip in percentage##

def bill():
    amount=eval(input("Enter the bill amount"))
    tip=eval(input("Enter the tip amount"))
    tip_per=(amount*tip)/100
    total=amount+tip_per
    print(f"the bill amount is {amount} and with tip is {total}")
bill()
```

```
the bill amount is 1000 and with tip is 1100.0
```

In [6]:
```
# create a function ask the user enter radius and calculate area of the circle#
import math
```

```
def circle():
    r=eval(input("Enter radius of the circle"))
    pi=math.pi
    area=round(pi*r*r,2)
    print(f"area of the circle is {area}")
circle()
```

```
area of the circle is 113.1
```

In [7]:
```
#create a fuction ask the user breadth and height and find area of triangle#

def triangle():
    h=eval(input("Enter height of the triangle"))
    b=eval(input("Enter breadth of the triangle"))
    area=round(0.5*b*h,2)
    print(f"Area of triangle is {area}")
triangle()
```

```
Area of triangle is 12.0
```

In [40]:
```
#create a function ask the user enter a number and find it even or odd##

def number():
    n=eval(input("Enter a number"))
    if n%2==0:
        print(f"{n} is even")
    else:
        print(f" {n} is odd")
number()
```

```
4 is even
```

In [42]:
```
# create a function ask the user enter a number find +ve or -vr or 0##

def ():
    n=eval(input("Enter a number"))
    if n>0:
        print(f"{n} is +ve")
    elif n<0:
        print(f"{n} is -ve")
    else:
        print(f"{n} is zero")
number()
```

```
5 is +ve
```

In [43]:
```
##create a function and find greater among three numbers##

def greatest():
    n1=eval(input("Enter a number1"))
    n2=eval(input("Enter a number2"))
    n3=eval(input("Enter a number3"))
    if n1>n2 and n1>n3:
        print(f"{n1} is big number")
    elif n2>n3:
        print(f"{n2} is big")
    else:
        print(f"{n3} is big")
number()
```

```
7 is big
```

Method-2

function with arguments

- how many variables are there in function
  - input variables:we will provide
  - suppose n1 and n2 are two input variables
  - output variables:we are creating a new variable using input variable

CONCENTRATE ON INPUT VARIABLES

In [44]:
```
def addition(n1,n2):
    add=n1+n2
    print(f"{add}")
addition(10,20)
```

```
30
```

In [9]:
```
def average(n1,n2,n3):
    avg=round((n1+n2+n3)/3,2)
    print(f"Average of {n1},{n2},{n3} is {avg}")
average(10,56,200)
```

```
Average of 10,56,200 is 88.67
```

In [ ]:
```
##perform all examples as assignment 4 ##
```

In [4]:
```
import random
def average(n1):
    print("n1:",n1)
    n2=eval(input("Enter a number2:"))
    n3=eval(input("enter a number3:"))
    avg=round((n1+n2+n3)/3,2)
    print(f"Average of {n1},{n2},{n3} is {avg}")
average(n1=random.randint(10,30))
```

```
n1: 23
Average of 23,5,6 is 11.33
```

In [6]:
```
def average(n1,n2):
    print("n1:",n1)
    print("n2:",n2)
    n3=eval(input("enter a number3:"))
    avg=round((n1+n2+n3)/3,2)
    print(f"Average of {n1},{n2},{n3} is {avg}")
average(10,40)
```

```
n1: 10
n2: 40
Average of 10,40,5 is 18.33
```

In [7]:
```
def average(n1):
    print("n1:",n1)
```

```
    n2=eval(input("Enter a number2:"))
    n3=eval(input("enter a number3:"))
    avg=round((n1+n2+n3)/3,2)
    print(f"Average of {n1},{n2},{n3} is {avg}")
average(10)
```

```
n1: 10
Average of 10,2689,456 is 1051.67
```

In [18]:
```
## create a function ask the user enter the bill and tip in percentage## three w

def bill(amount):

    tip=eval(input("Enter the tip amount"))
    tip_per=(amount*tip)/100
    total=amount+tip_per
    print(f"the bill amount is {amount} and with tip is {total}")
bill(1000)

import random
def bill(amount):
    print("amount",amount)
    tip=eval(input("Enter the tip amount"))
    tip_per=(amount*tip)/100
    total=amount+tip_per
    print(f"the bill amount is {amount} and with tip is {total}")
bill(random.randint(100,1000))


def bill(amount,tip):
    tip_per=(amount*tip)/100
    total=amount+tip_per
    print(f"the bill amount is {amount} and with tip is {total}")
bill(eval(input("enter amount")) ,eval(input("Enter the tip")))
```

```
the bill amount is 1000 and with tip is 1100.0
amount 117
the bill amount is 117 and with tip is 128.7
the bill amount is 1000 and with tip is 1100.0
```

In [19]:
```
def bill(amount):
    tip=eval(input("enter tip amount"))
    tip_per=(amount*tip)/100
    total=amount+tip_per
    print(f"the bill amount is {amount} and with tip is {total}")
bill(eval(input("enter amount")))
```

```
the bill amount is 1000 and with tip is 1100.0
```

In [22]:
```
import random
def bill(tip):
    amount=eval(input("enter amount"))
    print("tip",tip)
    tip_per=(amount*tip)/100
    total=amount+tip_per
    print(f"the bill amount is {amount} and with tip is {total}")
bill(random.randint(10,100))
```

```
tip 95
the bill amount is 1000 and with tip is 1950.0
```

Default arguments

- allreay fixed the arguments

In [23]:
```
def billing(bill,tip_per=20):
    print("The bill amount ",bill)
    print("The tip is:",tip_per)
    tip_amount=(bill*tip_per)/100
    total=bill+tip_amount
    print(total)
billing(1000)
```

```
The bill amount  1000
The tip is: 20
1200.0
```

case-1

- make all the arguments are default
- we no need to provide any values while calling a function

In [26]:
```
def billing(bill=1000,tip_per=20):
    print("The bill amount ",bill)
    print("The tip is:",tip_per)
    tip_amount=(bill*tip_per)/100
    total=bill+tip_amount
    print(total)
billing()
```

```
The bill amount  1000
The tip is: 20
1200.0
```

case-2

- All default parameters should assign after the non-default arguments

In [ ]:
```
#here non-default argument is after the default arguments it gives error#
#default argument must be last#
#when we call the function it assign the value for bill only not fir tip ,,,,so
def billing(bill=1000,tip_per):
    print("The bill amount ",bill)
    print("The tip is:",tip_per)
    tip_amount=(bill*tip_per)/100
    total=bill+tip_amount
    print(total)
billing(20)
```

In [27]:
```
def billing(tip_per,bill=1000):
    print("The bill amount ",bill)
    print("The tip is:",tip_per)
    tip_amount=(bill*tip_per)/100
    total=bill+tip_amount
    print(total)
billing(20)
```

```
The bill amount  1000
The tip is: 20
1200.0
```

case-3

- default arguments (values) might be over ride(it can change)

In [29]:
```
def billing(bill,tip_per=10):
    print("The bill amount ",bill)
    print("The tip is:",tip_per)
    tip_amount=(bill*tip_per)/100
    total=bill+tip_amount
    print(total)
billing(1000,20)
```

```
The bill amount  1000
The tip is: 20
1200.0
```

In [ ]:
```
def billing(bill=1000,tip_per=10):
    bill=5000
    print("The bill amount ",bill)
    print("The tip is:",tip_per)
    tip_amount=(bill*tip_per)/100
    total=bill+tip_amount
    print(total)
billing(2000,20)
```

In [ ]:
```
# step-1: define the function==1000

# step-2: call the function==2000  **while calling function only it runs**

#step-3: run the function==5000
```

In [30]:
```
def billing(bill=1000,tip_per=10):
    bill=5000                                ##step by step process##
    print("The bill amount ",bill)           #bill=1000
    print("The tip is:",tip_per)              #bill=7000
    tip_amount=(bill*tip_per)/100           # while calling bill=2000 ==runing
    total=bill+tip_amount
    print(total)
bill=7000

billing(2000,20)
```

```
The bill amount  5000
The tip is: 20
6000.0
```

Global variable and local variable

local variable

- local variable is defined inside the function
- local variable can not access outside the function
- global variable means outside the function

- global variable can access anywhere anytime

In [34]:
```
#**local variable**

def addition():
     n1=10
     n2=20
     print(n1+n2)
addition()
```

```
30
```

In [4]:
```
#global variable

number1=10
number2=10
def addition():
    add=number1+number2
    print(f"addition of {number1},{number2} is {add}")
addition()
```

```
addition of 10,10 is 20
```

In [5]:
```
number1
```

Out[5]: 10

In [6]:
```
number2
```

Out[6]: 10

In [7]:
```
#global variable


def addition():
    number_1=10
    number_2=80
    add=number1+number2
    print(f"addition of {number1},{number2} is {add}")
addition()
```

```
addition of 10,80 is 90
```

In [ ]:

In [ ]:

In [ ]: