FUNCTIONS,SCOPING AND ABSTRACTION

WHY FUNCTIONS?

- reusability
- compact and easy code to understand
- modularity

defiding function

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

In [3]:
```
greet??
```

calling function

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

```
hello world
```

function specification

- it include function name,parameters(if any) , and return type.
- ideally ,it mean include docstring

In [23]:
```
def add(x,y): # parameter
    """
    this function adds two number.

    parameter :
    - x(int):the first number
    - y(int):the second number

    return
    int: the sum of two number x and y
    """
    result = x+y
    print(f"sum is:",result)
```

calling function

add(1,20) ## 1 and 20 are the argument

In [45]:
```
## Acceing the docstring

print(add.__doc__)
```

```
this function adds two number.

parameter :
- x(int):the first number
- y(int):the second number

return
int: the sum of two number x and y
```

1.No parameter and No return type

In [24]:
```
def printline():
    s = input("enter the name = ")
    print(s)
printline()
```

```
enter the name = sds
sds
```

2.with parameter and No return type

In [30]:
```
def printline(s):
        print(s)

m = input("enter the name = ")

printline(m)
```

```
enter the name = sds
sds
```

3.with parameter and with return type

In [32]:
```
def printline(s):
        print(s)

m = input("enter the name = ")
t=printline(m)
print(t)
```

```
enter the name = sds
sds
None
```

4.No parameter and with return type

In [35]:
```
def printline():
    m = input("enter the name = ")
    return m

t=printline()
print(t)
```

```
enter the name = sds
sds
```

wrt function evenodd to check whether given no is even or odd

In [38]:
```
def evenodd():
    s=int(input("enter number"))
    if s%2==0:
        print("even number",s)
    else:
        print("odd number",s)
evenodd()
```

```
enter number5
odd number 5
```

returning multiple values

In [42]:
```
def sumsub(x,y):
    sum=x+y
    sub=x-y
    return sum,sub
sumsub(8,9)
```

Out[42]: (17, -1)

In [43]:
```
x,y=sumsub(8,9)
print(x)
print(y)
```

```
17
-1
```

defind a function a calculator that return addition,sub,multi,divison and **

In [44]:
```
def calc(x,y):
    sum=x+y
    sub=x-y
    mul=x*y
    expo=x**y
    div=x//y
    return sum,mul,expo,sub,div
calc(4,5)
```

Out[44]: (9, 20, 1024, -1, 0)

function parameter and argument

- a parameter is a variable name listed in the function parenthesis
- an argument is the value passed through the parameter when you call the function

default argument

In [46]:
```
def square(x=20):
    return x*x
print(square())# default
print(square(10))
```

```
400
100
```

positional argument

- the number of argument and position of argument must be matched.
- if we changed the order ,the result may changed.
- if we change the number of argument , we will get error

In [49]:
```
def sub(x,y):
    return x-y
print(sub(100,200))
print(sub(200,100))
# print(sub(100,200,300))
```

```
-100
100
```

keyword Argument

- we can pass the argument values by keyword name . the order does not matter.
- we can use postional and keyword argument simultaneously.
- the strict order is first positional and the keyword argument

Type *Markdown* and LaTeX: α^2

In [51]:
```
def wish(name,msg):
    print("hello",name,msg)
wish(name='python',msg="how are you?")
wish(msg="how are you?",name='python')
```

```
hello python how are you?
hello python how are you?
```

second case

In [53]:
```
wish("c++","goodmorning")
wish("c++",msg="goodmorning")
```

```
hello c++ goodmorning
hello c++ goodmorning
```

third case

In [54]:
```
wish(name="go","good afternoon")
```

```
  File "<ipython-input-54-a18b70a89c72>", line 1
    wish(name="go","good afternoon")
                   ^
SyntaxError: positional argument follows keyword argument
```

variable length argument

In [58]:
```
def sum(*n):
    total=0
    for i in n :
        total+=i
    print("the sum is",total)
```

In [56]:
```
sum(10)
```

```
the sum is 10
```

In [59]:
```
sum(10,20)
```

```
the sum is 30
```

In [60]:
```
sum(10,20,30)
```

```
the sum is 60
```

global and variables

In [65]:
```
x = 5
def fun():
    x=100
    print(x)
```

In [64]:
```
fun()
```

```
5
```

In [70]:
```
x = 5
def fun():
    global x
    x=100
    print(x)
```

In [73]:
```
fun()
```

```
100
```

In [74]:
```
print(x)
```

```
100
```

python seraches through yhe scope using following rule

- lebg-

- local
- enclosed
- globle
- bultin

wrt to take a number input from the user and check whether it is with in range by defindig a function name numberrange

In [79]:
```
def numberrange():
    x=int(input("enter number"))
    if x>=0 and x<=100:
        print("your number in range",x)
    else:
        print("your number is out of range",x)
numberrange()
```

```
enter number1000
your number is out of range 1000
```

task

- input:123456
- output:623451

In [86]:
```
def task():
    num =input("enter num:")
    listn= list(num)
    listn[0],listn[-1]=listn[-1],listn[0]
    print(listn)
task()
```

```
enter num:123456
['6', '2', '3', '4', '5', '1']
```

In [98]:
```
# swap first and last
def swap(number):
    #find num of digit
    numdigit=0
    temp=number
    while temp>0:
        temp//=10
        numdigit+=1
    #handle one digit
    if numdigit<=1:
        return number
    #extract first and last
    f=number//(10**(numdigit-1))
    l=number%10
    #remove first and last digit
    numwithout=(number%(10**(numdigit-1)))//10
    #swaped number
    swapnum=l*(10**(numdigit-1))+ numwithout*10+f
    return swapnum
num=int(input("enter num"))
swap(num)
```

```
enter num123456
```

Out[98]:
```
623451
```

In [91]:
```
x = input('enter any num : ')
print(x[-1] + x[1:(len(x)-1)] +x[0])
```

```
enter any num : 123456
623451
```

In [ ]:
```

```