Function

- A FUNCTION IS A SET OF STATEMENTS THAT TAKE INPUT,DO SOME SPECIFIC COMPUTATIONS AND PRODUCES OUTPUT.
- FUNCTIONS ARE REUSABLE.

PARAMETERS

- A PARAMETER IS A VARIABLE USED TO DEFINE A PARTICULAR VALUE DURING A FUNCTION DEFINITION

Arguments:
An argument is a value passed to a function at the time of function calling.

```
Syntax:   def function_name(define parameters):
              statement1
              statement2
              statement3
              .
              .
              .
              statement n
          function_name(pass arguments)
```

In [1]:

```
def isEvenorOdd(n):  #n=9
    if(n%2 == 0):
        print(n,"is even")
    else:
        print(n,"is odd")

```

In [2]:

```
isEvenorOdd(9)
```

```
9 is odd
```

In [3]:

```
def add(a,b): #a=54,b=56
    print(a+b)

add(54,56)
```

```
110
```

In [4]:

```
def isEvenorOdd(n):  #n=9
    if(n%2 == 0):
        print(n,"is even")
    else:
        return n

isEvenorOdd(7)
```

Out[4]:

```
7
```

In [7]:

```
def priya():
    print(" i printed")
def sai():
    return "i returned"

f1 = priya()
f2 = sai()
print(f1)
print(f2)
```

```
 i printed
None
i returned
```

In [9]:

```
def priya():
    print(" i printed")
def sai():
    return "i returned"

priya()
print(sai())
```

```
 i printed
i returned
```

In [11]:

```
def floor():
    print(5)
def ceil():
    return 7

floor()
print(floor())

```

```
5
5
None
```

In [14]:

```
def floor():
    print(5)
def ceil():
    print(7.8)


print(floor())
print(ceil())
```

```
5
None
7.8
None
```

In [16]:

```
#write factorial program
def factorial(n):
    fact = 1
    if(n == 1):
        print(1)
    else:
        for i in range(1,n+1):
            fact *=i
        print("n factorial is : fact")

m = int(input())
factorial(m)
```

```
5
n factorial is : fact
```

Types of functions in python

1. without arguments & without return values
2. without arguments & with return value
3. witth arguments & without return value
4. with arguments & with return value

In [17]:

```
#without arguments & without return values

def addition():
    a,b = 5,3
    print(a+b)

addition()
```

```
8
```

In [18]:

```
# 2.without arguments & with return

def mul():
    a,b =8,9
    res = a*b
    return res

print(mul())
```

72

In [19]:

```
# 3. with arguments & without return value

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

mul(89,10)
```

890

In [21]:

```
# 4.with argument & with return value

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

print(mul(5,9))
```

45

TYPES OF ARGUMENTS

1. Actual arguments
2. Formal arguments
3. Actual arguments
    A. position
    B. key
    C. defalut
    D. variable length arguments

In [ ]:

```
# actual arguments()
 add(5,6)
```

In [22]:

```
#1.positional arguments

def person(name,age):  # name = "priya" ,age = 20
    print("person name:",name)
    print("person age:",age)

person("priya",20)
```

```
person name: priya
person age: 20
```

In [24]:

```
def person(name,age):
    print("person name:",name)
    print("person age:",age-1)

person(20,"xyz")
```

```
person name: 20
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-24-057d5c499465> in <module>
      3     print("person age:",age-1)
      4
----> 5 person(20,"xyz")

<ipython-input-24-057d5c499465> in person(name, age)
      1 def person(name,age):
      2     print("person name:",name)
----> 3     print("person age:",age-1)
      4
      5 person(20,"xyz")

TypeError: unsupported operand type(s) for -: 'str' and 'int'
```

In [ ]:

```

```