Functions

- AFunction is a set of statements that take inputs,do some specific computation and produces output.
- Function are reusable.

Parameters: A parais 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
              .
              .
              statementn
            function_name(pass arguments)
```

In [1]:

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

In [2]:

```
isEvenorOdd(5)
```

```
5 is odd
```

In [4]:

```
n = 5
if(n%2 == 0):
    print(n,"is even")
else:
    print(n,"is odd")
```

```
5 is odd
```

In [5]:

```
def add(a,b):  #a=2,b=3
    print(a+b)

add(2,3)
```

```
5
```

In [6]:

```
def isEvenorOdd(n):  #n=5
    if(n%2 == 0):
        print(n,"is even")
    else:
        return n

isEvenorOdd(5)
```

Out[6]:

```
5
```

In [7]:

```
def fun that_prints():
    print("I printed")
def fun_that returns():
    return "I returned"

f1 = fun_that_prints()
f2 = fun_that_returns()
print(f1)
print(f2)
```

```
  File "<ipython-input-7-a146b70c46e1>", line 1
    def fun that_prints():
                  ^
SyntaxError: invalid syntax
```

In [11]:

```
def fun_that_prints():
    print("I printed")
def fun_that_returns():
    return "I returned"


print(fun_that_prints())
print(fun_that_returns())
```

```
I printed
None
I returned
```

In [12]:

```
def fun_that_prints():
    print("I printed")
def fun_that_returns():
    return "I returned"


fun_that_prints()
print(fun_that_returns())
```

```
I printed
I returned
```

In [16]:

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

```
5
5
None
7
```

In [22]:

```
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: 120
```

Types of functions in python

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

In [19]:

```
# Without arguments & without return values

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

Addition()
```

```
8
```

In [21]:

```
# 2.Without arguments & with return value

def Multiplication():
    a,b = 5,3
    res = a*b
    return res

print(Multiplication())
```

```
15
```

In [23]:

```
# with arguments & without return value
def Multiplication(a,b):
    print(a*b)

Multiplication(2,3)
```

```
6
```

In [25]:

```
# with arguments & with return value

def Mul(a,b):
    res = a*b
    return res

print(Mul(5,4))
```

```
20
```

Types of arguments

1. Actual arguments
2. Formal arguments
3. Actual arguments
    A. Position
    B. Keyword
    C. Default
    D. Variable length arguments

In [26]:

```
def add(a,b):
    c = a+b
    print(c)

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

```
11
```

In [29]:

```
# 1.Positional arguments

def person(name,age):
    print("Person name:",name)
    print("Person age:",age)

person("xyz",20)
```

```
Person name: xyz
Person age: 20
```

In [28]:

```
# 1.Positional arguments

def person(name,age):
    print("Person name:",name)
    print("Person age:",age)

person(30,"xyz")
```

```
Person name: 30
Person age: xyz
```

In [30]:

```
# keyword arguments
def person(name,age):
    print("Person name:",name)
    print("Person age:",age-1)

person(age=30,name="xyz")
```

```
Person name: xyz
Person age: 29
```

In [31]:

```
# Default argument
def person(name,age=21):
    print("Person name:",name)
    print("Person age:",age-1)

person("xyz")
```

```
Person name: xyz
Person age: 20
```

In [32]:

```
# 4.Variable Length argument
def add(a,b):
    print(a+b)

add(1,2,3,4)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-32-a10b6b34fde8> in <module>
      3     print(a+b)
      4
----> 5 add(1,2,3,4)

TypeError: add() takes 2 positional arguments but 4 were given
```

In [33]:

```
def add(a,*b):
    print("a=",a)
    print("b=",b)

add(1,2,3,4)
```

```
a= 1
b= (2, 3, 4)
```

In [35]:

```
def add(a,*b):
    s = a
    for i in b:
        s += i
    print(s)

add(1,2,3,4)
```

```
10
```

In [ ]:

```

```