

User-Defined Functions

- 1) Keyword `def` is used to start the Function Definition. `Def` specifies the starting of Function block.
- 2) `def` is followed by function-name followed by parenthesis.
- 3) Parameters are passed inside the parenthesis. At the end a colon is marked.

```
In [1]: def greet():
        print("Good morning")
        print("Guten Morgen")
        print("Shubh prabhat")

In [2]: greet()

Good morning
Guten Morgen
Shubh prabhat
```

Call a function in python

```
In [3]: def greet (name):
        print('Hello',name)
        print('How are you?')

In [4]: greet("Neha")

Hello Neha
How are you?
```

Example of return statement

- 1)`return[expression]` is used to send back the control to the caller with the expression.
- 2)`In case no expression is given after return it will return None.`
- 3)`In other words return statement is used to exit the Function definition.`

```
In [5]: def greet (name):
        print ('Hello',name)
        return
        print('How are you?')

In [6]: greet("Rahul")

Hello Rahul

In [7]: import builtins
marks=[92,89,85,81,72,63,68,59,61,55,60]
```

Find out average marks and grades

```
In [8]: import builtins
marks=[92,89,85,81,72,63,68,59,61,55,60]

In [9]: len(marks)

Out[9]: 11

In [10]: builtins.sum(marks)

Out[10]: 785

In [11]: total=builtins.sum(marks)

In [12]: average_marks=total/len(marks)

In [13]: average_marks

Out[13]: 71.36363636363636

In [14]: def compute_grade(average_marks):
        if average_marks >= 80.0:
            grade = 'A'
        elif average_marks >= 60:
            grade = 'B'
        elif average_marks >= 50:
            grade = 'C'
        else:
            grade = 'D'
        return grade

In [15]: grade =compute_grade(average_marks)

print("Your average marks is", average_marks)
print("Your grade is", grade)

Your average marks is 71.36363636363636
Your grade is B

In [16]: list_1=[2,4,8,10,12,14,16,18,20]

def cube(n):
    return n**3

print(list(map(cube, list_1)))

[8, 64, 512, 1000, 1728, 2744, 4096, 5832, 8000]

In [17]: list_1=[3,5,7,9,12,15,18,21,23]

def squareit(n):
    return n**2

print(list(map(squareit, list_1)))

[9, 25, 49, 81, 144, 225, 324, 441, 529]

In [18]: def sum(a,b):
        return(a+b)
sum(12,14)

Out[18]: 26

In [19]: number1=69
number2=89

In [20]: def multiply_numbers(num1, num2):
        return num1 * num2

In [21]: product_result = multiply_numbers(number1, number2)
print("Product of 69 and 89 is", product_result)

Product of 69 and 89 is 6141

In [22]: def factorial(n):
        fact=1
        for i in range(1,n+1):
            fact=fact*i
        return fact

In [23]: factorial(9)

Out[23]: 362880
```

BMI calculator using a User-Defined Function

```
In [24]: name1="Sudhanshi"
height_mt1=1.6
weight_kg1 = 60

name2="Adwait"
height_mt2=1.2
weight_kg2 = 25

name3="Rohith"
height_mt3=1.8
weight_kg3=85

In [25]: def bmi_calculator(name,height_mt,weight_kg):
        bmi=weight_kg/ (height_mt** 2)
        print("bmi: ")
        print(bmi)
        if bmi<25:
            return name + " is not overweight"
        else:
            return name + " is overweight"

In [26]: result1 =bmi_calculator(name1,height_mt1,weight_kg1)
result2 =bmi_calculator(name2,height_mt2,weight_kg2)
result3 =bmi_calculator(name3,height_mt3,weight_kg3)

bmi:
23.437499999999996
bmi:
17.361111111111111
bmi:
26.234567901234566

In [27]: print(result1)
print(result2)
print(result3)

Sudhanshi is not overweight
Adwait is not overweight
Rohith is overweight
```

Anonymous function/ Lambda

- 1)`Lambda functions are single-line functions defined without a name.`
- 2)`Anonymous function are created by using a keyword Lambda.`
- 3) `Lambda takes any number of arguments and returns an evaluated expression.`

```
In [28]: f= lambda a,b,c,d,e: a+b-c+d*e/2

In [29]: print(f(2,6,8,10,12))

60.0

In [30]: def sum(a,b,c,d,e):
        return (a+b-c+d*e/2)

In [31]: print(sum(2,6,8,10,12))

60.0

Lambda function to return the larger among two other numbers.

In [32]: x= lambda a,b :a if a>b else b
x(19,17)

Out[32]: 19

In [33]: sum=lambda x:x+6
print(sum(4))

10

In [34]: minus=lambda x,y:x-y//2
print(minus(89,59))

60

In [35]: square=lambda n: n*n
numbers=square(3)
print(numbers)

9
```

Sort the Students names based on the length

```
In [36]: names=["Priya","Ali","Honey", 'Arjun', 'Aradhna', 'Prashahti', 'Rohith', 'Sudhanshi', 'sahastrabuddhe', 'Bhattacharya']
names.sort(key=lambda names:len(names))
print(names,len(names))

['Ali', 'Priya', 'Honey', 'Arjun', 'Rohith', 'Aradhna', 'Prashahti', 'Sudhanshi', 'Bhattacharya', 'sahastrabuddhe'] 10

Program to filter out only the even items from a list
```

```
In [37]: my_list = [1, 5, 4, 6, 8, 11, 3, 12 ,20 ,33 ,68]

new_list = list(filter(lambda x: (x%2 == 0) , my_list))

print(new_list)

[4, 6, 8, 12, 20, 68]
```

Program to triple each item in a list using map()

```
In [38]: my_list = [1, 5, 4, 6, 8, 11, 3, 12]

new_list = list(map(lambda x: x * 3 , my_list))

print(new_list)

[3, 15, 12, 18, 24, 33, 9, 36]
```

Program to double each item in a list using map()

```
In [39]: my_list = [1, 5, 4, 6, 8, 11, 3, 12]

new_list = list(map(lambda x: x * 2 , my_list))

print(new_list)

[2, 10, 8, 12, 16, 22, 6, 24]

In [ ]:
```