**FUNCTIONS IN PYTHON**

**FUNCTION :** A function is a block of code that performs a specific task.

* Functions are defined by using the def keyword which follows by function name and parentheses which may have parameters.
* Any function can accept zero or more parameters.
* The body of the function contains the code that performs a specific task.
* Function returns a value using the return keyword, this makes a function to pass data back to the code from where it is called.
* To execute any function you need to call it by the function name followed by parentheses.

**Example :**

def squareroot(x): #defines a function named squareroot

return x \* x # returning the value

result = squareroot(5) #calling a function

print(result) # Output: 25

**Types of functions:**

In python there are different types of functions, now you will see some of them

1. Built-in functions
2. User defined functions
3. Lambda functions (Anonymous functions)

**Built-in Functions:**

Functions whose functionality is predefined. These are stored in the interpreter and come into action when you call them.

These can be accessed from any part of the code.

**Examples:** print ( ), len ( ), max ( ), set ( ).

**User defined functions:**

Functions that are defined by the user to perform a specific task.

Users can define their own function using the def keyword followed by function name.

**Lambda functions:**

Lambda functions are small, anonymous functions defined using the lambda keyword.

They can have any number of arguments but can only contain a single expression.

**ARGUMENTS :**

Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.

There are several ways to pass arguments to a function, each providing different capabilities and flexibility.

**Positional arguments:**

The values passed as arguments are assigned to parameters in the function definition in the same order.

It is the most common way of passing arguments to functions in Python.

Example:

Input :

def greet(name, age):

print("Hello,", name, "You are", age, "years old.")

#function with positional arguments

greet("Vinod", 30)

Output :

Hello, Vinod You are 30 years old.

**Keyword Arguments:**

Keyword arguments are passed with their corresponding parameter names, allowing for more flexibility in the order of arguments.

using the format parameter\_name=value in the function call.

Example:

Input:

def greet(name, age):

print("Hello,", name, "You are", age, "years old.")

#function with keyword arguments

greet(age=28, name="Vinod")

Output:

Hello, Vinod You are 28 years old.

**Variable length positional arguments**

It allows a function to take any number of positional arguments.

If the exact number of arguments are unknown then arbitrary arguments can be used,it can be done by adding “\*” before the argument makes it an arbitrary argument.

This way the function will receive a *tuple* of arguments, and can access the items accordingly:

**Example :**

Input:

def greet(\*names):

for name in names:

print("Hello,", name)

# Calling the function with variable-length positional arguments

greet("Lokesh", "Chan", "Ram")

Output :

Hello, Lokesh

Hello, Chan

Hello, Ram

“\*” indicates that it is an arbitrary argument and it can be accessed the same way as a tuple.

**RETURN VALUES :**

return keyword is used to return the values from a function

**Example :**

def cal\_area(length, width):

area = length \* width

return area

rectangle\_area = cal\_area(10, 9)

print("Area of the rectangle:", rectangle\_area)

Output :

Area of the rectangle: 90

When returned from a function it has to be collected in the variable and that variable can be used in the above example it's area.

**TYPES OF WAYS TO USE FUNCTIONS :**

1. No return, No arguments
2. No return, with arguments
3. Returns value, No arguments
4. Return value, with arguments

**1. No Return , No arguments**

A function with no parameters and no return value, a simple function that performs a specific task without taking any input parameters and without returning any result.

For example, consider the function “greet( )” which is used to print “Hello”, it does not return anything and does not take any arguments.

**Example :**

Input :

def greet():

print("Hello")

greet()

Output :

Hello

**2. No Return, With Arguments**

A function with parameters but no return value, it accepts input data through parameters but does not explicitly return any result.

For example, Consider a function “greet()” used to greet a person by a name which is taken as input.

**Example :**

Input **:**

def greet(name):

print("Hello,", name)

greet("Vamsi") # Call the function

Output :

Hello, Vamsi

**3. Returns Value, No Arguments**

A function that returns a value but takes no arguments takes after the execution of the function .

For example, Consider a function “get\_random\_number()” that takes no arguments and returns a random value between 1 to 100.

**Example :**

Input **:**

def get\_random\_number():

import random

return random.randint(1, 100)

random\_number = get\_random\_number()

print("Random number:", random\_number)

Output :

Random number: 95

**4. Returns Value, With Arguments**

A function that returns a value and also accepts arguments, it is a common type of function.

For example, Consider a function “power()” used to return the power of the number

**Example :**

Input **:**

def power(base, exponent):

result = base \*\* exponent

return result

result = power(5, 4)

print("Result:", result)

Output :

Result: 625