NPN Training

Training is the essence of success and we are committed to it.

PYTHON - Fuctions

Naveen P.N

Topics for the Module

After completing the module, you will be able to understand:

- Introduction to Functions
- Variable Arguments
- Returning from functions
- Global expressions

Introduction to functions & Function Definition

- Functions are self-contained blocks of code to perform a specific task .
- Functions are used for the following reasons :

1) Help reduce code redundancy 2)Help increase code re-usability 3)Improves program clarity

Syntax:

```
def function_name(parameters):
    function_body
    ...
```

Function call

- A function call is identified by the parantheses() that immediately follow the function name.
- A function name without the parantheses is a reference to the function and as such can be stored in variables and used later as a replacement for the original function name.

```
def hi()
    print(“hello world”)

>>> hi()
```

Positional Arguments

- The values that are sent to a function as part of the function call are called arguments.
- The values received by the function in it’s definition are called parameters.

</>

Hands on(Arguments_parameters.py)

Default Arguments

- Default argument values are values that are assumed to be present if not provided explicitly.
- It can be overridden by providing it explicitly like any other argument .

Hands on(Default_Arguments.py)

Keyword Arguments

- Keyword Arguments are special arguments where the parameter name is identified at the place of call along with the value..
- The ** prefix is used to denote there will be additional keyword arguments which are not present in the formal parameter list.

Hands on(Keyword_Arguments*.py)

Variable Arguments

- A function in python can be designed to receive any number of arguments .
- The number of arguments can vary from call to call.
- The special prefix * is used to denote that we are dealing with variable arguments

</>

Hands on(Variable_Arguments*.py)

Global Variables

- ❏ Global variables are variables that are accessible throughout the script -including inside functions as well as outside of all functions.

</>

Hands on(Global_Variables.py)

Key Takeaways

Hard work beats talent
when talent fails to **work hard**.