Python Programming

Functions

- Function is a named set of code that is wrapped up into a single unit
- Built-in functions
  - Pre-defined functions like len(),int(),input()
- Functions defined in modules
  - These functions are predefined in particular modules and can be used when the corresponding module is imported
- User defined functions
  - These functions are defined by the programmer

Defining Functions

- Syntax for defining a function is
  - def <function name>([parameters]):
    - statement(s)
- Function header
  - Begins with the keyword def and ends with a colon(:)
  - Specifies the name of the function and its parameters
- Function Body
  - The indented statements beneath the function header that defines the action performed by the function. The function body may or may not return a value

Defining Functions

- Statements which are not part of any function are called top level statements
- Python gives a special name to top level statements as __main__
- Function definitions are given at the top above all top level statements

Arguments and Parameters

- Values being passed through function call are called arguments(actual arguments)
- Values being received in function definition are called parameters(formal parameters)
- Arguments in Python can be:
  - Literals
  - Variables
  - expressions

Parameters

- Python supports 3 types of arguments / parameters
- Positional arguments(Required arguments)
  - Arguments must be provided for all parameters
  - The values of arguments are matched with parameters position wise
- Default arguments
  - The default values of parameters are specified in the function header of function definition
  - Non default parameters cannot follow default parameters
  - A parameter having a default value in function header becomes optional in function call.

Parameters

- Keyword(named) arguments
  - These are the named arguments with assigned values being passed in the function call statement.
- In a function call statement
  - an argument list must first contain positional arguments followed by any keyword argument.
  - default arguments should be the last in the argument list in the function call statement

return

- Non-void functions are functions returning some value
  - return statement returns a value from the function and exits the function
  - The returned value of a function should be used in the caller function
  - The return statement ends a function execution.
- Syntax is
  - return <value>
- The value being returned can be a literal, a variable or an expression

return

- Void functions are functions not returning any value
  - Void functions may or may not have a return statement.
  - If a void function has a return statement , it takes the following form
    - return

Returning multiple values

- To return multiple values from a function
  - The return statement should be of the form
    - return <value1>,<value2>,…….
  - The function call statement should receive or use the returned values
    - Receive the returned values in the form of a tuple
    - Specify the same number of variables on the left side of the assignment in function call.

Scope

- Parts of program within which a variable is legal and accessible is called scope of the variable
- There are broadly 2 kinds of scopes in Python:
  - Global Scope
    - A variable declared in top level segment(__main__) of a program is said to have global scope and is usable inside the whole program and all functions contained within the program
  - Local Scope
    - A variable declared in a function body is said to have local scope
    - It can be used only within the function in which it is declared

Name Resolution

- Variable in Global scope but not in local scope
- Variable neither in local scope nor in global scope
- Same variable name in global scope as well as local scope

Lambda Function

- A lambda function is an anonymous function defined without a name and without using the def keyword.
- A lambda function can take any number of arguments but can have only one expression.
- Syntax is

  lambda arguments: expression

Map Function

- The map function takes another function and a sequence of iterables as parameters and returns the output after applying the function to each item in the sequence.
- Syntax is

  map(function,iterable)

List Comprehension

- List comprehension allows you to create a new list with compact syntax and is based on the values of an existing list.
- It is defined inside square brackets and executes each element along with the for loop to iterate over each element.
- Syntax is

```
newlist=[expression (element) for element in oldlist if
                condition]
```

Thank you