# III B.Tech.

## Computer Science & Engineering

### CSE304: PYTHON PROGRAMMING WITH WEB FRAMEWORKS

### UNIT – I: Functions and Modules

**By**
**Mrs. S. KAMAKSHI, AP-III / CSE**
**School of Computing**

# Functions

- Reusable unit of code that performs a task
- Accepts zero or more arguments and returns one or more values
- Parameter passing order need not be same as the order specified in the function definition if they are passed using argument names
- Number of arguments passed to a function need not be same as the number of arguments specified in the function definition if they have proper packing and unpacking of arguments
- A function may or may not return a value
- The number of values returned from a function need not be same as the number of variables specified in the call of the function if they have proper packing and unpacking of values

# Default Arguments

- Assigning default values to one or more arguments
  - Name of the argument = value
- All default arguments must be placed at the end of argument list
- Syntax:

```
def fname(argument_list, default_arg = value):
        statements …
```

- Eg.

```
def f1(a, b, c=3):
    return a+b+c
```

- Call of function:

```
f1(10, 20, 30)
f1(1, 2)
f1(b=8, a=20)
```

# Named Arguments

- In function call, passing values using name
  - Name of the argument = value
- When passing values as named arguments, the order of arguments need not be in order.
- Syntax:

  fname(argument_name = value)
- Eg.

```
def f1(a, b, c):
    return a+b+c
```

- Call of function:

```
f1(a=10, c=20, b=30)
f1(b=1, c=2, a=3)
f1(b=8, a=20, c=12)
```

# Scope and Visibility of Objects

- Categories
  - Local
  - Non-local
  - Global
  - Built-in
- Built-in Objects
  - Objects defined in Python
- Global Objects
  - Objects defined outside of all functions
  - Accessed by any function  defined within the modules
  - Accessed by any function in  any other module in which it is imported
- Local Objects
  - Objects declared within a function
  - Accessible only within the function
- Non-local Objects
  - Objects declared in a function that encloses another function
  - Accessed by the enclosing function

# Modules

- A module is a file that stores reusable code
- For importing a module
  - Syntax:
    - import module_name [as namespace]
    - from module_name import fun_1 [, fun_2] … [as name]
    - from module_name import *
  - When importing a module, it is stored in a namespace specified
  - If namespace is not given, it will be imported into a namespace same as the name of the module
- When one or more functions are imported from a module it is stored in a namespace specified
  - If namespace is not given, it will be imported to the global namespace
- If two modules having same function name imported in to the global namespace, a name collision occurs.