Python Modules

Module

Modules allows to reuse one or more function in the program, even in the programs in which those functions have not been defined.

A module is .py extension file which has the definition of all the functions and variables that we can use in other programs.

The program which wants to use functions and variables defined in the module will use ***import keyword*** to link or import the module or the .py file.

Module Loading and Execution

- Python First Searches in the current working directory.
- Second search is in the directory specified in the PYTHONPATH environment variable.
- Lastly searches the Python installation path.

If not found anywhere then Import Error exception is generated.

How to Create a Module?

To create a module use .py extension to save your code in a file.

How to use a Module?

The **import** statement is used to import all the functionality of the module into another python program.

Example

The *from...import* Statement

```
from math import sqrt

print(sqrt(8))
```

Variables in Module

A module can also contain variables like arrays, dictionaries, objects etc.

Example.

Renaming a Module

"*as*" keyword can be used to rename a Module.

Example

Using dir() function

dir() is a built-in function which lists the identifiers defined inside a module.

These identifiers may be functions, classes and variables.

Scope of Variables

```
#module1

def new(x):

return x*3

#module2

def new(x):

return x**3

import module1

import module2

ans=new(10)
```

Scope of Variable

Each module in Python belongs to a **namespace**. This namespace includes name of all the function and variables defined inside a module.

When a program executes, three namespaces are referred:

I. built-in namespace

II. Local namespace

III. Global namespace

Scope of Variable

Built-in namespace contains name of all the built-in functions and constant.

Local namespace contains variables declared inside the function.

Global namespace contains all the identifiers declared inside a module.

Python interpreter when it encounters an identifier it first searches the local namespace and after that the global name space and finally built-in name space.

Private Module

By default, all the identifiers defined inside a module are public i.e. they are accessible by any module importing them.

In python two underscore __ can be placed before an identifier to make it a private identifier.

Python Packages

A Package is a hierarchical file directory structure.

It consists of modules and other packages within it.

Every package in Python is a directory which must have a special file called **_init_.py**

init.py file indicates that the directory contains a Python Package.

Python Package

Let's create a package named pack in the home directory. Consider the following steps.

1. Create a directory with name pack on path /**home**.

2. Create a python source file with name a.py and b.py on the path /**home**/Students.

3. To make this directory a package, we need to include one more file here, that is __init__.py which contains the import statements of the modules defined in this directory.

Python Package

Now, the directory **Pack** has become the package containing a python modules. Here we must notice that we must have to create __init__.py inside a directory to convert this directory to a package.