Unit-7 Modules and Directories

Modules

A Python module is a file containing Python definitions and statements. A module can define functions, classes, and variables. A module can also include runnable code. Grouping related code into a module makes the code easier to understand and use. It also makes the code logically organized.

Consider a module to be the same as a code library.

A file containing a set of functions you want to include in your application.

There are certain inbuilt and user defined modules (i.e. we can inbuilt the modules whenever it is required.

Frequently Inbuilt modules are as listed below: os module random module math module time module sys module collections module statistics module

We will discuss in detail os module in detail. Also a brief discussion about math module will be discussed.

Custom Modules can also be created using user defined.

Packages

A python package is a collection of modules. Modules that are related to each other are mainly put in the same package. When a module from an external package is required in a program, that package can be imported and its modules can be put to use. Any Python file, whose name is the module's name property without the .py extension, is a module.

A package is a directory of Python modules that contains an additional **init**.py file, which distinguishes a package from a directory that is supposed to contain multiple Python scripts. Packages can be nested to multiple depths if each corresponding directory contains its own **init**.py file.

Create a Module

To create a module just save the code you want in a file with the file extension .py:

Steps for creating a new module:

1. Go to jupyter notebook then create a new text file. Write module program

2. Save the above text file in .py format
3. Go to jupyter notebook new python kernel. And then write import module code in new notebook.
4. Run the program and see the output.

In [7]:
```
#1. Go to jupyter notebook then create a new text file. Write module program
#Example-Save this code in a file named modpract.py

def greeting(name):
    print("Hello," + name)
```

2. Save the above text file in .py format

Use a Module Now we can use the module we just created, by using the import statement:

In [5]:
```
# 3. Go to jupyter notebook new python kernel. And then write import module code in ne
#Example import the module named mymodule, and call the greeting function:

import modpract
modpract.greeting("Jonathan")
```

```
Hello,Jonathan
```

The difference between import and from import in Python is:

1. import imports the whole library.
2. from import imports a specific member or members of the library.

When Use 'import' and When 'from import'

Use from import when you want to save yourself from typing the module name over and over again. In other words, when referring to a member of the module many times in the code.

Use import when you want to use multiple members of the module.

In [15]:
```
##Example of import module math
import math
x = 10
y = 4
d = math.sqrt(x ** 2 + y ** 2)
r = 5.0
area = math.floor(math.pi * r ** 2)
print(area)
```

```
78
```

In [16]:
```
##Example of from module import function*
from math import sqrt, floor
x = 5.2
y = 2.4
d = floor(sqrt(x ** 2 + y ** 2))
print(d)
```

```
5
```

In [1]:
```
##Example of from module import function*
from math import *
x = 5.2
y = 2.4
d = floor(sqrt(x ** 2 + y ** 2))
print(d)
```

```
5
```

Python-OS Module:

It is possibly to automatically perform many operating system tasks.

The OS module in Python provides functions for craeting and removing a directory (folder), fetching it's contents, and changing and identifying the current directory,etc.

You first need to import os statement before using it's functions.

Getting current working directory:-

The getcwd() function confirms the current working directory.

In [18]:
```
#Example for getcwd() for getting current working directory
#Program to get the following functions: getcwd(), chdir(), mkdir(),listdir(),remove()
#import operating system
import os
#to get current working directory
print(os.getcwd())
```

```
C:\Users\Abhi
```

Creating a Directory:-

We can create a new directory using the os.mkdir() function as shown below:

In [20]:
```
#Example to create a directory in Python using mkdir()
#Program to get the following functions: getcwd()
#import operating system
import os
#to create new directory
os.mkdir("D:\Pyt")
#Open the folder in D Drive, and check MyPythonProjects folder will be created
print("Directory created")
```

```
Directory created
```

In [22]:
```
#Program to get the following functions: getcwd(), mkdir()
#import operating system
import os
#to create new directory
print(os.getcwd())
print(os.mkdir("Make"))
#As path is not specified so,open the folder in C Drive, and Check path i.e. In C driv
#then go to Abhi check a folder named My and MyPythonProgram will be created
```

```
C:\Users\Abhi
None
```

Removing a Directory

The rmdir() function in the OS module removes the specified directory either with an absolute or relative path. Note that, for a directory to be removed, it should be empty.

In [24]:
```
#Program to get the following functions: rmdir()
#import operating system
import os
# to remove directory
os.rmdir("D:\\Pyt")
#Check the D Drive Folder the folder will be deleted
```

List Files and Sub-directories:

The listdir() function returns the list of all files and directories in the specified directory.

If we don't specify any directory, then list of files and directories in the current working directory will be returned.

In [25]:
```
#Program to get the following functions: listdir()
#import operating system
import os
#to list directory
print(os.listdir())
#As path is not specified then by default the directory will be taken and will list ev
```

```
['.idlerc', '.ipynb_checkpoints', '.ipython', '.jupyter', '.matplotlib', '3D Object
s', 'AppData', 'Application Data', 'Contacts', 'Cookies', 'demofile.txt', 'demofile2.
txt', 'Desktop', 'Documents', 'Downloads', 'employee.py', 'Favorites', 'IGC', 'Immuta
ble Data Structures.ipynb', 'Links', 'Local Settings', 'Make', 'Matplotlib.ipynb', "M
CQ'sQB.ipynb", 'Music', 'My', 'My Documents', 'mymodule.ipynb', 'mymodule.py', 'MyP
y', 'NetHood', 'NTUSER.DAT', 'ntuser.dat.LOG1', 'ntuser.dat.LOG2', 'NTUSER.DAT{53b39e
88-18c4-11ea-a811-000d3aa4692b}.TM.blf', 'NTUSER.DAT{53b39e88-18c4-11ea-a811-000d3aa4
692b}.TMContainer00000000000000000001.regtrans-ms', 'NTUSER.DAT{53b39e88-18c4-11ea-a8
11-000d3aa4692b}.TMContainer00000000000000000002.regtrans-ms', 'ntuser.ini', 'OneDriv
e', 'P-1.ipynb', 'P-2.ipynb', 'Pictures', 'Practicals.ipynb', 'PrintHood', 'ProgramsQ
B.ipynb', 'Recent', 'Saved Games', 'ScStore', 'Searches', 'SendTo', 'Start Menu', 'Te
mplates', 'Unit 1 Introduction to Python and Jupyter Notebooks.ipynb', 'Unit- 6 Worki
ng with Files.ipynb', 'Unit-5 Mutable Data Structure.ipynb', 'Unit-7 Modules and Dire
ctories.ipynb', 'Untitled.ipynb', 'Untitled1.ipynb', 'Untitled2.ipynb', 'Untitled3.ip
ynb', 'Untitled4.ipynb', 'Untitled5.ipynb', 'Untitled6.ipynb', 'Untitled7.ipynb', 'Vi
deos', '__pycache__']
```

chdir() Change Directory method using OS Module

os.chdir() method in Python used to change the current working directory to specified path. It takes only a single argument as new directory path.

Syntax: os.chdir(path) Parameters: path: A complete path of directory to be changed to new directory path. Returns: Doesn't return any value

In [1]:
```
#Program to get the following functions: chdir()
#import operating system
import os
# change the current directory
# to specified directory
os.chdir("D:\\MyPyth123")
print(os.getcwd())

print("Directory changed")
```

```
D:\MyPyth123
Directory changed
```

os.remove()

os.remove() method in Python is used to remove or delete a file path. This method can not remove or delete a directory.

In [33]:
```
#Python program to explain os.remove() method

# importing os module
import os

#First create a new text file in drive

# File name
file = 't2.txt'
```

```
# File location
location = "C:\\Users\Abhi"

# Path
path = os.path.join(location, file)

# Remove the file
# 't1.txt'
os.remove(path)
print("%s has been removed successfully" %file)
```

```
t2.txt has been removed successfully
```