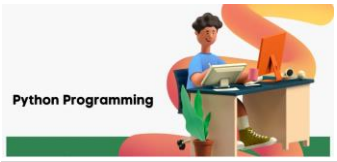
LECTURE TEN

Python Directory and Files Management

- A directory is a digital structure used to store and organize files on a computer.
- A directory is a collection of files and subdirectories. A directory inside a directory is known as a subdirectory.
- Python has the os module that provides us with many useful methods to work with directories (and files as well).



Current work Directory in Python

We can get the present working directory using the **getcwd()** method of the os module. This method returns the current working directory in the form of a string.

Example: Current work Directory in Python
import os print(os.getcwd())

Changing Directory in Python

In Python, we can change the current working directory by using the **chdir()** method.

Example: Changing Directory in Python

import os
os.chdir('C:\\Python99') # change directory
print(os.getcwd())

List Directories and Files in Python

All files and sub-directories inside a directory can be retrieved using the **listdir()** method.

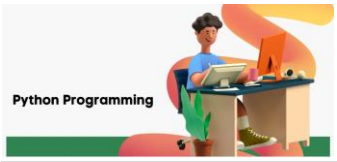
Example: List Directories and Files in Python
import os print(os.listdir())

Renaming a Directory or a File

The rename() method can rename a directory or a file.

Example: Renaming a Directory or a File in Python

import os
os.rename('test','new_one') # rename directory
print(os.listdir())



Making a New Directory in Python

In Python, we can make a new directory using the mkdir() method. To create a nested directory structure (such as a directory inside another directory), use the **os.makedirs()** method.

Example: Making a New Directory in Python

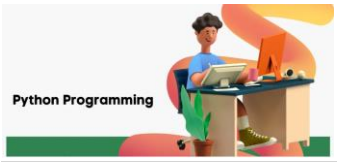
```
import os
os.mkdir('test')            # make directory
path = 'projects/games/game01'
os.makedirs(path)
print(os.listdir())
```

Delete Directory or File in Python

To delete Directory or File, you must import the OS module, and run its **os.remove().**

Example: Delete Directory or File in Python

```
import os
os.remove("demo.txt")          # delete " demo.txt" file
os.rmdir("mydir")              # delete empty directory "mydir"
import shutil
shutil.rmtree("mydir")   #delete "mydir" directory and all of its content
```



Check if File exist in Python

To avoid getting an error, you might want to check if the file exists before you try to delete it:

Example: Check if File exist in Python

```
import os
if os.path.exists("demo.txt"):
   os.remove("demo.txt")
else:
  print("The file does not exist")
```

Joining and Splitting Path in Python

join() in python joins path components and returns a path as a string. It adds appropriate separators (\ for Windows and / for Unix). Conversely, split() splits the path into components, removing the separator.

Example: Joining and Splitting Path in Python

```
import os
os.path.join('C:','Users', 'msm','Desktop')
os.path.split('C:Users\\ msm \\Desktop')
```

Recursively Traversing a Directory in Python

The walk() function lets us recursively traverse a directory. This means that it returns the roots, subdirectories, and files in a directory.You can traverse it using for loops in Python.

Example: Recursively Traversing a Directory in Python

```
import os
for roots,dirs,files in os.walk('C:\\Users\\'msm'\\Desktop\\Papers'):
            print(roots,len(dirs),len(files))
```