# Assignment 10 Solutions

#### 1. How do you distinguish between shutil.copy() and shutil.copytree()?

**Ans:** shutil.copy() method is used to copy the contents of a file from one file to another file/folder, it primary takes two arguments src,dest, src represents the file to be copied where as destination refers to the file/folder to where the src data should be copied, if dest is a folder name the src with exact name will be copied to the dest folder, if its a file then the contents of src will be copied to dest where dest retains it name.

shutil.copytree() function is used to copy the entire contents of a folder to other folder. it also takes two arguments src & dest, it copies all the content recursively and stores it in dest. the important catch here is dest must not exist prior to this and it will be created during the copy operation. Permissions and times of directories are copied with shutil.copystat() and individual files are copied using shutil.copy2() by default which can be modified using copy\_function attribute.

#### 2. What function is used to rename files??

**Ans:** os.rename() function is used to rename files or directories using a python program, this function takes two arguments src and dest, src represents the name file/directory which we want to rename, whereas dest represents the new name of the file/directory.

#### 3. What is the difference between the delete functions in the send2trash and shutil modules?

**Ans:** Shutil module provides a funciton called as shutil.rmtree() which deletes a directory and all its contents. The other functions with similar functionality are os.remove() -> removes a file, os.rmdir() removes a empty directory. The problem with these functions is once a file is deleted. it will be lost permanently, if a file is deleted accidentally using these methods there is no way we can recover the deleted file

Where as send2trash module provides a function called send2trash.send2trash() to delete a file/directory. these methods moves the files/directories to trash folder instead of permanently deleting them. hence if a file/folder is deleted accidentally it can be still recovered from trash folder, if is deleted using the send2trash.send2trash() function. send2trash is not included with python standard libary like os & shutil modules. it needs to be installed explicitly using the command !pip install send2trash

#### 4.ZipFile objects have a close() method just like File objects’ close() method. What ZipFile method is equivalent to File objects’ open() method?

**Ans:** ZipFile Module provides a method called as zipfile.ZipFile() to read and write to zipFiles. it takes arugments lile filename and mode etc zipfile.ZipFile('filename', mode = 'r')

#### 5. Create a programme that searches a folder tree for files with a certain file extension (such as .pdf or .jpg). Copy these files from whatever location they are in to a new folder

In [1]:

**import** os

**import** shutil

**def** search\_and\_copy(source,destination,extensions):

source **=** os**.**path**.**abspath(source)

destination **=** os**.**path**.**abspath(destination)

**for** foldername, subfolder, filenames **in** os**.**walk(source):

print(f'Folder Name ➞ {foldername}',end**=**'\n\n')

print(f'Sub Folders ➞ {subfolder}',end**=**'\n\n')

print(f'Files ➞ {filenames}',end**=**'\n\n')

**for** filename **in** filenames:

fileName,extension **=** os**.**path**.**splitext(filename)

**if** extension **in** extensions:

targetFile **=** foldername**+**os**.**path**.**sep**+**fileName**+**extension

shutil**.**copy(targetFile, destination)

print(f'Files copied successfully from {source} to {destination}')

extensions **=** ['.pdf','.jpg','.ipynb']

source **=** 'Dummy Source'

destination **=** 'Dummy Destination'

search\_and\_copy(source, destination, extensions)

Folder Name ➞ C:\Users\vishnu.adepu\Documents\iNeuron-Assignments\Python Basic Assignment\Dummy Source

Sub Folders ➞ []

Files ➞ ['01.Assignment\_01.ipynb', '02.Assignment\_02.ipynb', '03.Assignment\_03.ipynb', '04.Assignment\_04.ipynb', '05.Assignment\_05.ipynb', '06.Assignment\_06.ipynb', '07.Assignment\_07.ipynb', '08.Assignment\_08.ipynb', '09.Assignment\_09.ipynb', '10.Assignment\_10.ipynb', '11.Assignment\_11.ipynb', '12.Assignment\_12.ipynb', '13.Assignment\_13.ipynb', '14.Assignment\_14.ipynb', '15.Assignment\_15.ipynb', '16.Assignment\_16.ipynb', '17.Assignment\_17.ipynb', '18.Assignment\_18.ipynb', '19.Assignment\_19.ipynb', '20.Assignment\_20.ipynb', '21.Assignment\_21.ipynb', '22.Assignment\_22.ipynb', '23.Assignment\_23.ipynb', '24.Assignment\_24.ipynb', '25.Assignment\_25.ipynb']

Files copied successfully from C:\Users\vishnu.adepu\Documents\iNeuron-Assignments\Python Basic Assignment\Dummy Source to C:\Users\vishnu.adepu\Documents\iNeuron-Assignments\Python Basic Assignment\Dummy Destination

**1.To what does a relative path refer ?**

**Ans:** The relative path is the path to some file with respect to your current working directory (PWD).  
**For example:** if Absolute path to a file called stuff.txt is: C:/users/admin/docs/stuff.txt If my PWD is C:/users/admin/ , then the relative path to stuff.txt would be: docs/stuff.txt  
**Note:** PWD + relative path = absolute path

**2.Where does an absolute path start with your Operating System ?**

**Ans:** In Linux based systems the absolute path starts with **/**. Where as in Windows based systems absolute path starts with **C:**

**3.What does the functions os.getcwd() and os.chdir() do ?**

**Ans:** os.getcwd() method tells us the location of current working directory (CWD). Whereas os.chdir() method in Python used to change the current working directory to specified path. These functions are similar to linux commands pwd and cd

In [1]:

**import** os

print(os**.**getcwd()) *# Prints the current Working Directory*

path **=** r'C:\Users\vishnu.adepu\Documents'

os**.**chdir(path)

print(os**.**getcwd())

C:\Users\vishnu.adepu\Documents\iNeuron-Assignments\Python Basic Assignment

C:\Users\vishnu.adepu\Documents

**4.What are . and .. folders ?**

**Ans:** . Represents the Current Directory Whereas .. Represents the Parent Directory of the Current Directory  
**For Example:** if the below path is my absolute path:  
C:\\Users\\vishnu\\Documents\\iNeuron-Assignments\\Python Basic Assignment  
Then . represents the path C:\\Users\\vishnu\\Documents\\iNeuron-Assignments\\Python Basic Assignment  
Where as .. represents the path C:\\Users\\vishnu\\Documents\\iNeuron-Assignments

**5.In C:\bacon\eggs\spam.txt which part is the dir name and which part is the base name ?**

**Ans:** For C:\bacon\eggs\spam.txt  
The dir name is C:\\bacon\\eggs  
The Base name is spam.txt

In [2]:

**import** os

path **=** r'C:\bacon\eggs\spam.txt'

print(os**.**path**.**dirname(path))

print(os**.**path**.**basename(path))

C:\bacon\eggs

spam.txt

**6.What are the three mode arguments that can be passed to the open() function ?**

**Ans:** A file can be Accessed in python using open() function. open function takes two arguments filename and mode of operation (optional). if mode is not provided the default mode of opening is read mode  
So, the syntax being: **open(filename, mode)**

* **‘r’** – Read Mode: This is the default mode for open(). The file is opened and a pointer is positioned at the beginning of the file’s content.
* **‘w’** – Write Mode: Using this mode will overwrite any existing content in a file. If the given file does not exist, a new one will be created.
* **‘r+’** – Read/Write Mode: Use this mode if you need to simultaneously read and write to a file.
* **‘a’** – Append Mode: With this mode the user can append the data without overwriting any already existing data in the file.
* **‘a+’** – Append and Read Mode: In this mode you can read and append the data without overwriting the original file.
* **‘x’** – Exclusive Creating Mode: This mode is for the sole purpose of creating new files. Use this mode if you know the file to be written doesn’t exist beforehand.

**7.What happens if an existing file is opened in write mode ?**

**Ans:** Using this mode will overwrite any existing content in a file. If the given file does not exist, a new one will be created.

**8.How do you tell the difference between read() and readlines() ?**

**Ans:** The main difference is that **read()** will read the whole file at once and then print out the first characters that take up as many bytes as you specify in the parenthesis

* Whereas the **readline()** that will read and print out only the first characters that take up as many bytes as you specify in the parenthesis. You may want to use readline() when you're reading files that are too big for your RAM.
* The **read()** would treat each character in the file separately, meaning that the iteration would happen for every character.
* The **readline()** function, on the other hand, only reads a single line of the file. This means that if the first line of the file were three lines long, the readline() function would only parse (or iterate/operate) on the first line of the file.

**9.What data structure does a shelf value resemble ?**

**Ans:** it contains key and values it represents dictionary.