**Assignment – 10**

Q1. How do you distinguish between shutil.copy() and shutil.copytree()?

Sol. In Python, the "shutil" module provides two functions for copying files and directories:

1. "shutil.copy()" and "shutil.copytree()". Here's how you can distinguish between them:

shutil.copy(): The "shutil.copy()" function is used to copy a single file from a source location to a destination location. It takes two arguments: the path of the source file and the path of the destination file. If the destination path does not exist, the function will create the necessary directories to complete the copy. However, if the destination path already exists, the function will overwrite the existing file with the source file.

Example usage:

import shutil

# Copy a single file

shutil.copy('source\_file.txt', 'destination\_file.txt')

2. shutil.copytree(): The "shutil.copytree()" function is used to copy an entire directory tree (including all files and subdirectories) from a source directory to a destination directory. It takes two arguments: the path of the source directory and the path of the destination directory. The destination directory must not already exist, as the function will create it along with any necessary subdirectories.

Example usage:

import shutil

# Copy an entire directory tree

shutil.copytree('source\_directory', 'destination\_directory')

To summarize:

shutil.copy() is used for copying individual files.

shutil.copytree() is used for copying entire directory trees, including files and subdirectories.

Q2. What function is used to rename files?

Sol. In Python, the "os" module provides a function called "os.rename()" that is used to rename files. It allows you to change the name of an existing file to a new name specified by you. The "os.rename()" function takes two arguments: the current name of the file and the new name you want to assign to it.

Example usage:

import os

# Rename a file

os.rename('old\_file.txt', 'new\_file.txt')

Using the "os.rename()" function, you can change the name of a file within the same directory. It does not move the file to a different location, only changes the name of the file.

Note: It's important to be cautious when renaming files, as it can potentially overwrite existing files if the new name conflicts with an existing file in the same directory.

Q3. What is the difference between the delete functions in the send2trash and shutil modules?

Sol.

The "send2trash" and "shutil" modules in Python provide different approaches for deleting files and directories. Here's how you can distinguish between their delete functions:

1. send2trash module: The "send2trash" module provides a function called "send2trash()" that moves files and directories to the system's trash or recycle bin instead of permanently deleting them. It acts as a safer alternative to the "os.remove()" function, as it allows for potential recovery of deleted items from the trash.

Example usage:

import send2trash

# Send a file or directory to the trash

send2trash.send2trash('file\_to\_delete.txt')

By using "send2trash.send2trash()", you can delete files or directories by moving them to the trash or recycle bin, providing an opportunity for recovery if needed.

1. shutil module: The "shutil" module provides a function called "shutil.rmtree()" that is used to remove (delete) an entire directory tree, including all files and subdirectories. Unlike "send2trash()", this function permanently deletes the specified directory and its contents, bypassing the system's trash or recycle bin.

Example usage:

import shutil

# Delete an entire directory tree

shutil.rmtree('directory\_to\_delete')

Q4. ZipFile objects have a close() method just like File objects’ close() method. What ZipFile method is

equivalent to File objects’ open() method?

Sol: The equivalent method in the ZipFile object to the open() method of File objects is the ZipFile constructor itself.

1. To open a ZIP file using the ZipFile object, you can create an instance of the ZipFile class by passing the path to the ZIP file as a parameter to the constructor. This is similar to how the open() function is used to open a file object.

Example usage:

import zipfile

# Open a ZIP file

zip\_file = zipfile.ZipFile('example.zip')

1. In the example above, zipfile.ZipFile('example.zip') creates a ZipFile object that represents the opened ZIP file named "example.zip". You can then use this zip\_file object to perform various operations on the ZIP file, such as extracting its contents, adding files to it, or obtaining information about the archived files.

Remember to close the ZipFile object using its close() method when you have finished working with the ZIP file, just like you would close a File object.

Example:

# Close the ZipFile object

zip\_file.close()

Q5. 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.

Sol.

**Code:**

import os

import shutil

def search\_and\_copy\_files(source\_folder, target\_folder, file\_extension):

for root, \_, files in os.walk(source\_folder):

for file in files:

if file.endswith(file\_extension):

source\_path = os.path.join(root, file)

target\_path = os.path.join(target\_folder, file)

shutil.copy2(source\_path, target\_path)

print(f"Copied: {source\_path} to {target\_path}")

# Example usage

source\_folder = 'path/to/source/folder' # Specify the source folder to search in

target\_folder = 'path/to/target/folder' # Specify the target folder to copy files to

file\_extension = '.pdf' # Specify the file extension to search for

# Call the function

search\_and\_copy\_files(source\_folder, target\_folder, file\_extension)

In the code above, the search\_and\_copy\_files() function takes three parameters: the source\_folder to search in, the target\_folder to copy the files to, and the file\_extension to search for. It recursively walks through the folder tree using os.walk(), checks each file's extension, and if it matches, copies the file using shutil.copy2() to the target folder.

Make sure to replace 'path/to/source/folder', 'path/to/target/folder', and '.pdf' with your desired folder paths and file extension.

Note: The shutil.copy2() function is used instead of shutil.copy() to preserve the original file's metadata, such as timestamp