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

**Answer:**

shutil.copy() and shutil.copytree() are two functions used for different purposes when copying files and directories.

**shutil.copy(src, dst):** This function is used to copy a single file from the source path (src) to the destination path (dst). It creates a new file at the destination with the same contents as the source file. If the destination file already exists, it will be overwritten.

**Example:**

**import shutil**

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

**hutil.copy()** is used to copy the file source\_file.txt to destination\_file.txt. If destination\_file.txt already exists, it will be replaced with the contents of source\_file.txt.

**shutil.copytree(src, dst):** This function is used to recursively copy an entire directory from the source path (src) to the destination path (dst). It creates a new directory at the destination and copies all the files and subdirectories from the source directory to the destination directory. If the destination directory already exists, an error will occur unless the dirs\_exist\_ok parameter is set to True.

**Example:**

**import shutil**

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

**shutil.copytree**() is used to copy the entire directory source\_directory to destination\_directory. If destination\_directory already exists, an error will occur.

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

**Answer:**

The function used to rename files is shutil.move(src, dst).

**The shutil.move()** function is versatile and can be used for both renaming files and moving files to different directories. It effectively performs a combination of the os.rename() and shutil.copy2() functions.

**Example:**

**import shutil**

**src = 'old\_filename.txt'**

**dst = 'new\_filename.txt'**

**shutil.move(src, dst)**

the file with the name **'old\_filename.txt'** will be renamed to **'new\_filename.txt' using shutil.move().** If the destination file already exists, it will be overwritten with the renamed file.

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

**Answer:**

The delete functions in the **send2trash** and **shutil modules, send2trash.send2trash()** and **shutil.rmtree(),** are used to delete files and directories in Python. However, there are differences in their behavior and the scope of deletion:

**send2trash.send2trash(path):** This function is provided by the send2trash module, which allows you to send files or directories to the operating system's trash or recycle bin instead of permanently deleting them. It provides a safer approach to deleting files as they can be recovered from the trash if needed.

**Example:**

**import send2trash**

**send2trash.send2trash('file.txt')**

**shutil.rmtree(path**): This function is part of the shutil module and is used to remove a directory and its contents recursively. It permanently deletes the specified directory and all its subdirectories and files. This function does not move the files to the trash; it directly removes them.

**Example:**

**import shutil**

**shutil.rmtree('directory')**

**send2trash.send2trash**() moves files or directories to the system's trash or recycle bin for possible recovery, while **shutil.rmtree()** permanently deletes files and directories without the possibility of recovery. The choice between them depends on whether you want a more reversible deletion (using send2trash) or a permanent deletion (using shutil.rmtree).

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

**Answer:**

The **ZipFile** method that is equivalent to File objects' **open()** method is **ZipFile()** itself.

In the zipfile module, you create a ZipFile object by calling its constructor with the path to the ZIP file you want to work with. This is similar to how you open a regular file using the open() function.

**Example:**

**import zipfile**

**# Open a ZIP file for reading with zipfile.ZipFile('example.zip', 'r') as zip\_file:**

**# Perform operations on the ZipFile object**

**file\_list = zip\_file.namelist()**

**# ...**

**# After the 'with' block, the ZipFile object is closed automatically**

In this example, **zipfile.ZipFile('example.zip', 'r')** creates a ZipFile object that opens the ZIP file named 'example.zip' in read mode ('r'). The resulting ZipFile object, zip\_file, can be used to perform various operations on the ZIP file, such as accessing the list of file names (namelist()).

the equivalent of the File objects' open() method for **ZipFile** objects is to **create a ZipFile obje**ct by calling **zipfile.ZipFile()** with the appropriate parameters.

**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.**

**Answer:**

**import os**

**import shutil**

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

**# Create the target folder if it doesn't exist**

**os.makedirs(target\_folder, exist\_ok=True)**

**# Walk through the source folder and its subdirectories**

**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"File copied: {source\_path} -> {target\_path}")**

**# Example usage**

**source\_folder = 'path/to/source/folder'**

**target\_folder = 'path/to/target/folder'**

file\_extension = '.pdf'

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

In this program, the function search\_and\_copy\_files() takes the source folder, target folder, and file extension as input. It creates the target folder if it doesn't exist already. Then, it traverses the source folder and its subdirectories using os.walk(), and for each file with the specified file extension, it copies it to the target folder using shutil.copy2().

You need to provide the appropriate paths for the source\_folder, target\_folder, and file\_extension variables according to your requirements. The shutil.copy2() function is used to preserve the original file's metadata, such as timestamps, during the copy process.

Make sure you have the necessary permissions to access the files in the source folder and to create files in the target folder.