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

Ans. The shutil module in Python provides functions for file and directory operations. Two commonly used functions in this module are shutil.copy() and shutil.copytree(), which are used for copying files and directories, respectively. Here's how you can distinguish between them:

**shutil.copy(src, dst)**: This function is used to copy a single file from the source (src) to the destination (dst). It takes two arguments: the path to the source file and the path to the destination file. If the destination file already exists, it will be overwritten. The copy() function preserves the file's metadata (such as permissions and timestamps) during the copy operation.

Example usage:

***import shutil # Copy a file from source to destination***

***shutil.copy('path/to/source/file.txt', 'path/to/destination/file.txt')***

**shutil.copytree(src, dst):** This function is used to recursively copy an entire directory tree from the source directory (src) to the destination directory (dst). It creates a new directory at the destination and copies all files and subdirectories from the source directory to the destination. If the destination directory already exists, a FileExistsError will be raised. The copytree() function preserves the directory structure and file metadata during the copy operation.

Example usage:

***import shutil # Copy an entire directory tree from source to destination shutil.copytree('path/to/source', 'path/to/destination')***

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

Ans. In Python, the os module provides a function called os.rename() that is commonly used to rename files. The os.rename() function allows you to change the name of a file by providing the current file path and the desired new file path.

Here's the basic syntax for using os.rename():

***import os***

***# Rename a file***

***os.rename(current\_file\_path, new\_file\_path)***

In the above code , cu**rrent\_file\_path** represents the current path of the file that you want to rename, and **new\_file\_path** represents the desired new path and name for the file. The os.rename() function will move and rename the file accordingly.

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

Ans. **send2trash module**: The send2trash module provides a function called send2trash(), which moves files or directories to the system's trash or recycle bin instead of permanently deleting them. It provides a safer alternative to os.remove() or os.rmdir() because it allows for easy recovery of deleted items from the trash.

The send2trash() function takes a path as an argument and moves the specified file or directory to the trash. It works on both files and directories, and it supports cross-platform usage.

Example usage:

***from send2trash import send2trash***

***# Move a file or directory to the trash***

***send2trash('path/to/file.txt')***

**shutil module:** The shutil module provides several functions for file and directory operations, including functions for deleting files and directories. The commonly used functions for deletion are shutil.rmtree() and os.remove().

shutil.rmtree() is used to delete a directory and its contents recursively. It removes the specified directory along with all its subdirectories and files.

Example usage:

***import shutil***

***# Delete a directory and its contents***

***shutil.rmtree('path/to/directory')***

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

Ans. The equivalent method to the open() method of file objects in the ZipFile class is the ZipFile() constructor itself. The ZipFile constructor is used to create a new ZipFile object, similar to how the open() function is used to open a file and create a file object.

Here's an example to illustrate the usage of the ZipFile() constructor to open a ZIP file:

***import zipfile***

***# Open a ZIP file using the ZipFile constructor***

***zip\_file = zipfile.ZipFile('path/to/archive.zip', 'r')***

***# Access the contents of the ZIP file***

***# Close the ZIP file***

***zip\_file.close()***

In the above example, the ZipFile() constructor is used to open the ZIP file located at 'path/to/archive.zip' in read mode ('r'). It returns a ZipFile object, which allows access to the contents of the ZIP file.

**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.**

**Ans.**

**import os**

**import shutil**

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

# Create the destination folder if it doesn't exist

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

# Recursively search the source folder

**for foldername, subfolders, filenames in os.walk(source\_folder):**

**for filename in filenames:**

**if filename.endswith(file\_extension):**

**source\_path = os.path.join(foldername, filename)**

**destination\_path = os.path.join(destination\_folder, filename)**

# Copy the file to the destination folder

**shutil.copy2(source\_path, destination\_path)**

**print(f"Copied: {source\_path} --> {destination\_path}")**

# Specify the source folder, destination folder, and file extension

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

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

**file\_extension = '.pdf'** # Change this to the desired file extension

# Call the function to search and copy files

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