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

`shutil.copy()` and `shutil.copytree()` are both functions provided by the Python standard library's `shutil` module for copying files and directories, respectively.

The main difference between `shutil.copy()` and `shutil.copytree()` is that `shutil.copy()` is used to copy a single file from one location to another, whereas `shutil.copytree()` is used to copy an entire directory tree (i.e., a directory and all of its contents, including subdirectories and files) from one location to another.

Here are some more specific differences between the two functions:

- `shutil.copy()` takes two arguments: the source file path and the destination file path. It will copy the file at the source path to the destination path. If the destination file already exists, it will be overwritten by the copied file.

- `shutil.copytree()` takes two arguments: the source directory path and the destination directory path. It will copy the entire directory tree at the source path to the destination path. If the destination directory already exists, `shutil.copytree()` will raise an error. However, you can use the optional third argument, `dirs\_exist\_ok=True`, to allow the destination directory to exist and simply add the new files and directories to it.

So, if you need to copy a single file, use `shutil.copy()`. If you need to copy an entire directory tree, use `shutil.copytree()`.

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

The `os.rename()` function is used to rename files in Python. It takes two arguments: the current file name and the new file name. For example, to rename a file named `old\_file.txt` to `new\_file.txt`, you would use the following code:

```python

import os

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

```

This function can also be used to move a file by providing a new directory path as the new file name argument.

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

The delete functions in the `send2trash` and `shutil` modules are used for different purposes.

The `send2trash` module provides a `send2trash()` function that moves a file or folder to the trash or recycle bin instead of permanently deleting it. This is a safer option than using the `os.remove()` function, which permanently deletes the file or folder.

On the other hand, the `shutil` module provides a `rmtree()` function that can be used to delete an entire directory tree, including all the files and subdirectories it contains. This function deletes the files and directories permanently, without sending them to the trash or recycle bin.

Therefore, the main difference between the delete functions in `send2trash` and `shutil` is that the former sends files and folders to the trash or recycle bin, while the latter permanently deletes them.

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

The equivalent method in the `ZipFile` class to the `open()` method in the `File` class is the `ZipFile()` constructor method.

The `ZipFile()` constructor method is used to create a new `ZipFile` object and open a ZIP archive file. It takes the filename of the ZIP archive as its first argument and a mode string as its second argument, which specifies the mode in which the ZIP archive should be opened (e.g., read, write, append, etc.).

Here's an example of how to create a new `ZipFile` object and open a ZIP archive file in read mode:

```

import zipfile

with zipfile.ZipFile('example.zip', 'r') as zip:

# do something with the ZIP archive

```

In this example, the `ZipFile()` constructor method is used to create a new `ZipFile` object and open the `example.zip` archive file in read mode (`'r'`). The `with` statement is used to ensure that the `ZipFile` object is automatically closed and any resources are cleaned up after the block of code is executed.

Once the `ZipFile` object is created, you can use its various methods to read, write, or modify the contents of the ZIP archive. When you're finished working with the ZIP archive, you can close the `ZipFile` object by calling its `close()` method, just like you would with a `File` object.

**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.**

import os

import shutil

# Set the directory to search for files

search\_dir = "/path/to/search/directory"

# Set the directory to copy the files to

copy\_dir = "/path/to/copy/directory"

# Set the file extension to search for

extension = ".txt"

# Create the copy directory if it doesn't exist

if not os.path.exists(copy\_dir):

os.makedirs(copy\_dir)

# Walk through the search directory and find files with the given extension

for root, dirs, files in os.walk(search\_dir):

for file in files:

if file.endswith(extension):

# Get the full path of the file to copy

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

# Create the destination directory if it doesn't exist

dest\_dir = os.path.join(copy\_dir, os.path.relpath(root, search\_dir))

if not os.path.exists(dest\_dir):

os.makedirs(dest\_dir)

# Copy the file to the destination directory

dest\_file\_path = os.path.join(dest\_dir, file)

shutil.copy2(file\_path, dest\_file\_path)