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

   In Python, both shutil.copy() and shutil.copytree() are methods in the shutil module used for file operations, but they have different purposes.

   shutil.copy(src, dst) is used to copy a single file from the source path src to the destination path dst. It preserves the metadata of the original file, such as the file permissions and timestamps. If the destination file already exists, it will be overwritten

   shutil.copytree(src, dst) is used to copy a directory tree from the source path src to the destination path dst. It recursively copies the entire directory tree and all its contents, including subdirectories and files, to the destination path. If the destination directory already exists, shutil.copytree() will raise a FileExistsError

   So, the main difference between shutil.copy() and shutil.copytree() is that the former is used to copy a single file while the latter is used to copy an entire directory tree. shutil.copy() is useful when we want to make a single copy of a file, while shutil.copytree() is useful when we want to duplicate an entire directory structure.

2. What function is used to rename files??

   In Python, the os.rename() function is used to rename files. The os module provides a way to interact with the file system, and the rename() function can be used to rename files by providing the current file name as the first argument and the new file name as the second argument

   ```
   import os

   os.rename('old_file_name.txt', 'new_file_name.txt')
   ```

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

   In Python, the send2trash and shutil modules provide functions for deleting files and directories, but they differ in their approach and behavior.

   The send2trash module provides a send2trash() function that sends files or directories to the operating system's trash or recycle bin instead of permanently deleting them. This means that the files or directories can be restored if needed. The send2trash() function moves the specified file or directory to the trash or recycle bin, depending on the operating system

   On the other hand, the shutil module provides the os.remove() and shutil.rmtree() functions for deleting files and directories, respectively. The os.remove() function deletes a single file, while the shutil.rmtree() function recursively deletes a directory and all its contents. These functions permanently delete the specified files or directories, and they cannot be restored

   So, the main difference between the send2trash and shutil delete functions is that the former sends the specified files or directories to the trash or recycle bin, while the latter permanently deletes them. The send2trash function provides a safety net by allowing files or directories to be restored if necessary, while the shutil functions permanently remove them from the system

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 ZipFile objects to File objects' open() method is ZipFile.open().

In the ZipFile module, open() method is used to access the contents of the files within a ZIP archive. This method opens the specified file in the archive and returns a file-like object that can be used to read its contents. Here is an example usage:

import zipfile

open a zip file and access its contents

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

```
    with myzip.open('file_inside_zip.txt') as myfile:

        content = myfile.read()

        print(content)
```

In this example, ZipFile is used to open the ZIP archive file my_zip_file.zip in read mode. The open() method is then used to open the file named file_inside_zip.txt within the archive. The file is then read and its contents are printed.

It's worth noting that the ZipFile module's open() method returns a file-like object, which means it provides a file-like interface for reading the contents of the file within the ZIP archive, but it is not the same as opening a regular file with the built-in open() function.

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 [2]:
```
import os
import shutil

# function to search for files with a given extension and copy them to a new folder
def copy_files_by_extension(src_folder, dest_folder, extension):
    # create the destination folder if it doesn't exist
    if not os.path.exists(dest_folder):
        os.makedirs(dest_folder)

    # walk through the directory tree and copy files with the given extension to the d
    for dirpath, dirnames, filenames in os.walk(src_folder):
        for filename in filenames:
            if filename.endswith(extension):
                file_path = os.path.join(dirpath, filename)
                shutil.copy(file_path, os.path.join(dest_folder, filename))

# Example usage
copy_files_by_extension('/path/to/source/folder', '/path/to/destination/folder', '.pdf
```

This program defines a function copy_files_by_extension that takes three arguments: the source folder to search for files, the destination folder to copy the files to, and the file extension to search for (e.g. .pdf).

The function first checks if the destination folder exists and creates it if it doesn't. Then, it uses os.walk to traverse the directory tree starting from the source folder. For each file in the tree, it checks if the file has the specified extension and if so, copies it to the destination folder using shutil.copy.

We can use this program by calling the copy_files_by_extension function with the appropriate arguments for specific use case.

In [ ]: