''' ********** FILE MANAGEMENT ON DOWNLOADS FOLDER PROGRAM **********

Make an application designed to improve the organisation of a computer's download folder.
.....The program creates folders for each format with the name of the file extension
.....after automatically classifying files based on their extensions.
.....The root folder is effectively cleaned using this technique, and
.....similar files are logically arranged into the proper directories.

********** Pseudocode **********

Step .....1.....Go for Import necessary libraries from os and glob
Step .....2.....Go for discover a list of files in the current directory in root of Downloads folder
Step .....3.....Create a set to store unique file extensions and storing the names of all files in the current directory
Step .....4.....Set up an empty set named extensions_set and initialize it with all possible file extensions.
Step .....5.....Go through each file in files_list iteratively:
Step .....5.1 ....Extract the file extension, which is the portion following the last dot, and include it in extensions_set.
Step .....6.....Go through to Set up the function create_folders for each unique extension
Step .....6.1.....Go for building an empty set named existing_folders for the purpose of keeping track of existing folders.
Step .....6.2.....Set a repeat this process for every extension in extensions_set.
step ....6.2.1....Verify if the file's extension is ".py" (Python script file); if it is, no move on to the following extension.\
Step ....6.2.2....Set a folder name based on the extension (for example, if the extension is "pdf," the folder_name will be "pdf_Files")
Step ....6.2.3....Verify that folder_name exists anywhere in the directory; if it does, move on to the next extension.
Step ....6.2.4....Useinf function create folder_name to create the folder.
Step ....6.2.5....Add existing folders using folder_name.
Step .....7.....Go for develop a move_files function:
Step .....7.1.....Set up to go over each file in files_list iteratively:
Step .....7.1.1...Set up extract the file extension, first.
Step .....7.1.2...Verify that the file's extension is ".py" (Python script file);
.....if it is, go on to the rest of the file.
Step .....7.1.3...Create a folder_name depending on the extension.
Step .....7.1.4...Verify that the folder_name is among the already_existing directories; if it is, place the file there.
Step .....8.....To create folders for each distinct extension, use the create_folders method.
Step .....9 ....To transfer files to the appropriate directories, use the move_files function.
Syep .....10.....Provide suitable error handling for folder existence, file not found, and permission problems.
Step .....11.....Setup 3 testing for each step of program and print Testing Successfully.

********** Start Program **********

Description .....Create a program that automatically organises your download

```
folder by file type,
.....................creating a folder for each type. It analyzes the folder,
extracts file extensions,
.....................searches for a folder with the same name, creates it if it
doesn't exist,
.....................and then transfers the files to the correct folde.
.....................This tool saves you time and effort by helping you to stay
organised, reduce clutter, and be more efficient.
Author ..............Mohammadreza Habibinejadnad Kochesfehani STUDENT ID: 1174672
Date ................01/09/2023,05/09/2023,08/09/2023,10/09/2023,11/09/2023,13/09
/2023, 15/09/2023
time ..............  6:15pm  ,  9:20pm  ,   5:30pm ,  10:30am ,   8:45am ,
6:15pm  ,   7:10pm
version ...  1.0  .....Made program and pseudocode, started coding.
version ...  2.0  .....Made functions for creating folders and moving files.
version .... 3.0  .....Added exception handling for open files and printed error
messages.
version .... 4.0  .....Used os.path.splitext to split the filename and extension,
which is more robust than splitting by '.'
.....................and handles filenames with multiple dots correctly.
.....................Added exception handling for folder creation and file
movement to handle possible errors more gracefully.
.....................Skipped '.py' files during folder creation and file
movement.
.....................Printed error messages for any exceptions that occur during
the process.
version .... 5.0  .....Avoided duplication for multiple runs of the program.
version .... 6.0  .....Control moving folder and check for existing folder if not
create a folder
version .... 7.0  .....Added testing for the program with 3 tests.
.....................Fixed the checking for if an extension's target folder
exists.
.....................Transferred the file to the root directory if the target
folder doesn't exist, and outputted relevant messages.
version .... 8.0 ......Improved the output in the terminal to print all activity of
the program and the test program.


*********************************      Stat Program Coding
*********************************************
'''
# Import necessary libraries
import os
import glob
import shutil

# Set this variable to True to enable testing
testing = True

# Discover a list of files in the current directory
files_list = glob.glob('*')

# Create a set to store unique file extensions
extensions_set = set()

# Iterate through each file in the list
for each_file in files_list:
    # Extract the extension from the file name
    extension = each_file.split('.')[-1]
```

```
    # Check if the file is a Python file (.py) or a folder
    if extension.lower() == 'py' or os.path.isdir(each_file):
        continue  # Ignore Python files and folders

    # Convert to lowercase for case-insensitive comparison
    extensions_set.add(extension.lower())

# If testing is enabled, print the extensions_set (Test 1)
if testing:
    print("Extensions Set:", extensions_set)
    print("Test 1 SUCCESSFUL.\n")

# Function to create folders for each unique extension using os.makedirs


def create_folders():
    existing_folders = set()  # Initialize an empty set for existing folders
    for item in os.listdir():
        if os.path.isdir(item):
            existing_folders.add(item)  # Add existing folder names to the set

    print("Existing folders before creating new ones:")
    for folder in existing_folders:
        print(folder)

    for ext in extensions_set:
        if ext == 'py':
            continue  # Skip Python files
        # Folder name based on file format (extension) + "_Files"
        folder_name = ext + "_Files"

        # Check if the folder already exists in the set of existing_folders
        if folder_name not in existing_folders:
            os.makedirs(folder_name)
            print(f"Created folder: {folder_name}")
            # Add the newly created folder to the set
            existing_folders.add(folder_name)


# If testing is enabled, run Test 2 (folder creation)
if testing:
    print("Running Test 2: Folder Creation")
    create_folders()

    # Check if the folders were created successfully
    folder_names = [ext + "_Files" for ext in extensions_set if ext != 'py']
    created_folders = [
        folder for folder in folder_names if os.path.exists(folder)]

    if len(created_folders) == len(folder_names):
        print("All folders created successfully:")
        for folder in created_folders:
            print(folder)
        print("Test 2 successful.\n")
    else:
        print("Test 2 failed. Some folders were not created.\n")

# Function to move files to their respective folders using shutil.move
```

```
def move_files():
    for each_file in files_list:
        # Extract the extension from the file name
        extension = each_file.split('.')[-1]
        if extension == 'py' or each_file.endswith('_Files'):
            continue  # Skip Python files and existing folders

        folder_name = extension + '_Files'

        # Check if the folder exists
        if os.path.exists(folder_name) and os.path.isdir(folder_name):
            try:
                shutil.move(each_file, folder_name + '/' + each_file)
                print(f"Moved file '{each_file}' to folder '{folder_name}'")
            except FileNotFoundError:
                print(f"File not found: {each_file}")
            except FileExistsError:
                print(f"File already exists in destination: {each_file}")


# If testing is enabled, run Test 3 (file moving)
if testing:
    print("Running Test 3: File Moving")
    move_files()

    # Check if files were moved successfully
    remaining_files = [file for file in files_list if os.path.exists(file)]

    if len(remaining_files) == 0:
        print("All files moved successfully.")
        print("Test 3 successful.")
    else:
        print("Test 3 is SUCCESSFUL if one Python file was not moved:")
        for file in remaining_files:
            print(file)


'''   *************              End Program              ****************'''
```