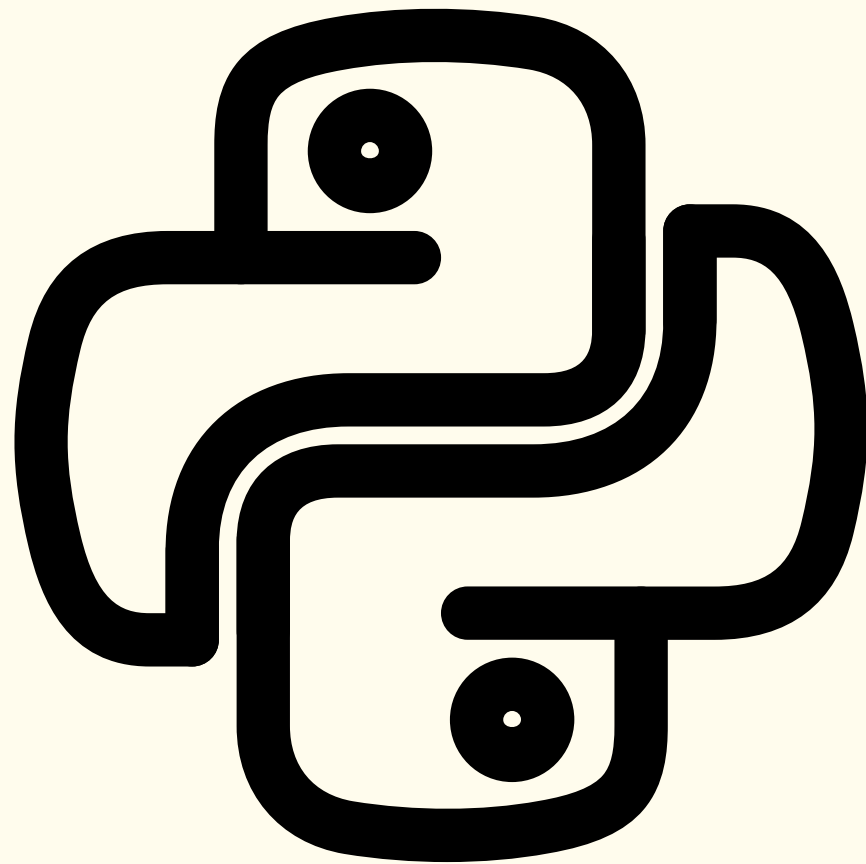


PROBLEM SOLVING WITH PYTHON

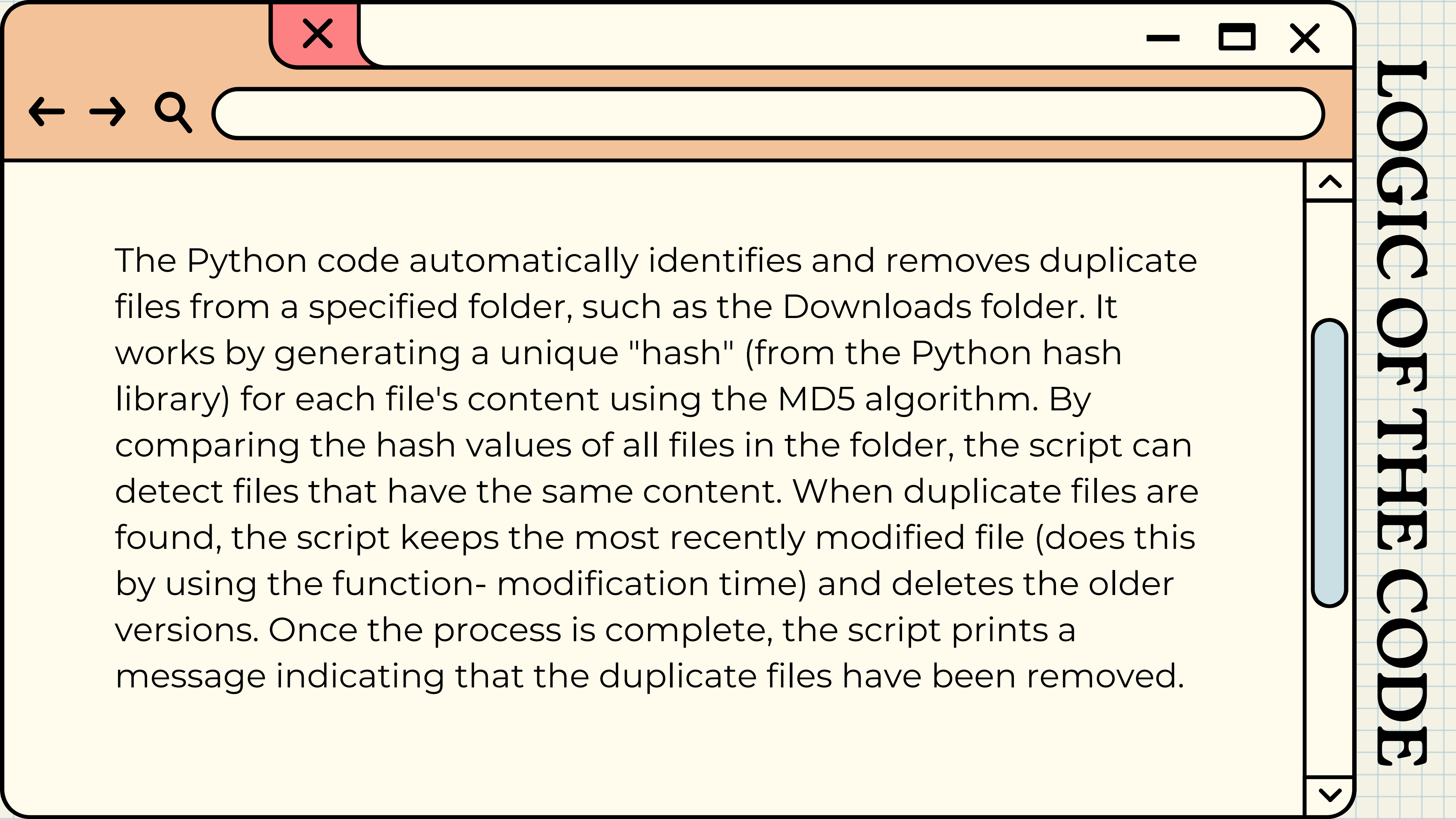


- **Objective: Apply Python programming to solve a design-related problem.**



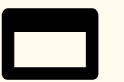
PROBLEM STATEMENT

- We often encounter the issue of accumulating duplicate files—documents, images, folders, etc.—in our storage. This typically happens when we accidentally download multiple copies of the same file and forget to delete the older versions, resulting in cluttered storage. To address this, I developed a code designed to automatically delete older duplicate files in the download folder while retaining only the latest modified versions making it convenient for the user to track and declutter the downloaded files.



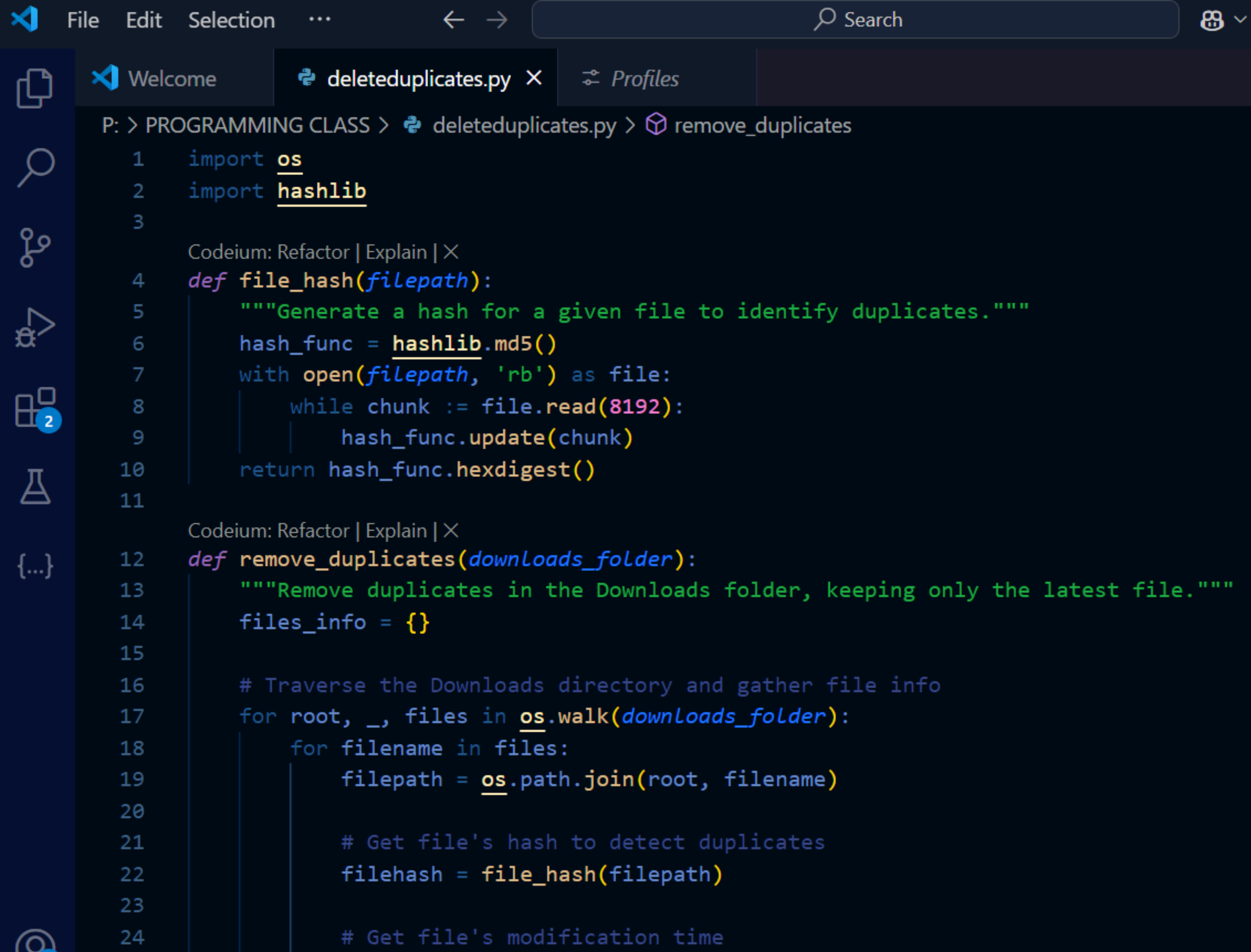
The Python code automatically identifies and removes duplicate files from a specified folder, such as the Downloads folder. It works by generating a unique "hash" (from the Python hash library) for each file's content using the MD5 algorithm. By comparing the hash values of all files in the folder, the script can detect files that have the same content. When duplicate files are found, the script keeps the most recently modified file (does this by using the function- modification time) and deletes the older versions. Once the process is complete, the script prints a message indicating that the duplicate files have been removed.

PROBLEMS FACED



In terms of what I was able to accomplish, I successfully ensured that all duplicate files in the Downloads folder were deleted, except for the latest modified version. However, I wanted to add a feature that would either automatically rename the latest file as "file name-latest/modified" or prompt the user with the option to rename it. One thing I couldn't achieve was having the program run automatically as soon as a duplicate file was downloaded. Instead, I managed to schedule the task to run at a specific time each day using Windows Task Scheduler.

This way, I no longer need to manually run the program each time I want to remove duplicates—it now runs automatically on my local device at the set time every day. Setting this up was a bit challenging, as Windows required several permissions to allow the Python code to interact with and modify system files.



```
P: > PROGRAMMING CLASS > deleteduplicates.py > remove_duplicates

1  import os
2  import hashlib
3
4  Codeium: Refactor | Explain | X
5  def file_hash(filepath):
6      """Generate a hash for a given file to identify duplicates."""
7      hash_func = hashlib.md5()
8      with open(filepath, 'rb') as file:
9          while chunk := file.read(8192):
10             hash_func.update(chunk)
11         return hash_func.hexdigest()
12
13  Codeium: Refactor | Explain | X
14  def remove_duplicates(downloads_folder):
15      """Remove duplicates in the Downloads folder, keeping only the latest file."""
16      files_info = {}
17
18      # Traverse the Downloads directory and gather file info
19      for root, _, files in os.walk(downloads_folder):
20          for filename in files:
21              filepath = os.path.join(root, filename)
22
23              # Get file's hash to detect duplicates
24              filehash = file_hash(filepath)
25
26              # Get file's modification time
```

THE CODE

```
12 def remove_duplicates(downloads_folder):
24     # Get file's modification time
25     file_mod_time = os.path.getmtime(filepath)
26
27     # If the hash already exists, compare modification times
28     if filehash in files_info:
29         existing_file, existing_mod_time = files_info[filehash]
30
31         # Ensure both files are in the Downloads folder
32         if filepath.startswith(downloads_folder) and existing_file.startswith(downloads_folder):
33             if file_mod_time > existing_mod_time:
34                 # Keep the latest file, remove the older one
35                 os.remove(existing_file)
36                 files_info[filehash] = (filepath, file_mod_time)
37             else:
38                 os.remove(filepath)
39         else:
40             # Store file info if it's the first occurrence of the hash
41             files_info[filehash] = (filepath, file_mod_time)
42
43     print("Duplicate files in the Downloads folder have been removed.")
44
45 # Example usage:
46 downloads_folder_path = os.path.expanduser('~Downloads') # Path to the Downloads folder
47 remove_duplicates(downloads_folder_path)
```