

```
"""
```

Python File Handling & Automation Script

Author: Your Name

Description:

This script demonstrates:

1. Reading and writing text files
2. Reading and writing CSV files
3. Automating file operations (rename, move, delete)
4. Using try-except for error handling

```
"""
```

```
import os      # For file & folder operations
```

```
import csv     # For CSV file handling
```

```
import shutil  # For moving files
```

1. WRITE TO A TEXT FILE

```
try:
```

```
    # Open file in write mode (creates file if not exists)
```

```
    with open("sample.txt", "w") as file:
```

```
        file.write("Hello, this is a sample text file.\n")
```

```
        file.write("Python File Handling Example.\n")
```

```
    print("Text file created successfully.")
```

```
except Exception as e:
```

```
print("Error while writing text file:", e)
```

2. READ FROM TEXT FILE

```
try:
```

```
    with open("sample.txt", "r") as file:
```

```
        content = file.read()
```

```
        print("\nReading Text File:")
```

```
        print(content)
```

```
except FileNotFoundError:
```

```
    print("Text file not found.")
```

```
except Exception as e:
```

```
    print("Error while reading text file:", e)
```

3. WRITE TO CSV FILE

```
try:
```

```
    with open("students.csv", "w", newline="") as csvfile:
```

```
        writer = csv.writer(csvfile)
```

```
        # Writing header
```

```
        writer.writerow(["ID", "Name", "Marks"])
```

```
        # Writing data rows
```

```
        writer.writerow([1, "Karan", 85])
```

```
        writer.writerow([2, "Rahul", 90])
```

```
writer.writerow([3, "Sneha", 88])
```

```
print("CSV file created successfully.")
```

```
except Exception as e:
```

```
    print("Error while writing CSV file:", e)
```

4. READ FROM CSV FILE

```
try:
```

```
    with open("students.csv", "r") as csvfile:
```

```
        reader = csv.reader(csvfile)
```

```
        print("\nReading CSV File:")
```

```
        for row in reader:
```

```
            print(row)
```

```
except FileNotFoundError:
```

```
    print("CSV file not found.")
```

```
except Exception as e:
```

```
    print("Error while reading CSV file:", e)
```

5. AUTOMATION - RENAME FILE

```
try:
```

```
    os.rename("sample.txt", "renamed_sample.txt")
```

```
    print("\nFile renamed successfully.")
```

```
except FileNotFoundError:
```

```
    print("File to rename not found.")
```

```
except Exception as e:
```

```
    print("Error while renaming file:", e)
```

6. AUTOMATION - MOVE FILE

```
try:
```

```
    # Create folder if not exists
```

```
    if not os.path.exists("backup"):
```

```
        os.makedirs("backup")
```

```
    # Move file to backup folder
```

```
    shutil.move("renamed_sample.txt", "backup/renamed_sample.txt")
```

```
    print("File moved to backup folder.")
```

```
except FileNotFoundError:
```

```
    print("File to move not found.")
```

```
except Exception as e:
```

```
    print("Error while moving file:", e)
```

7. AUTOMATION - DELETE FILE

```
try:
```

```
    os.remove("students.csv")
```

```
    print("CSV file deleted successfully.")
```

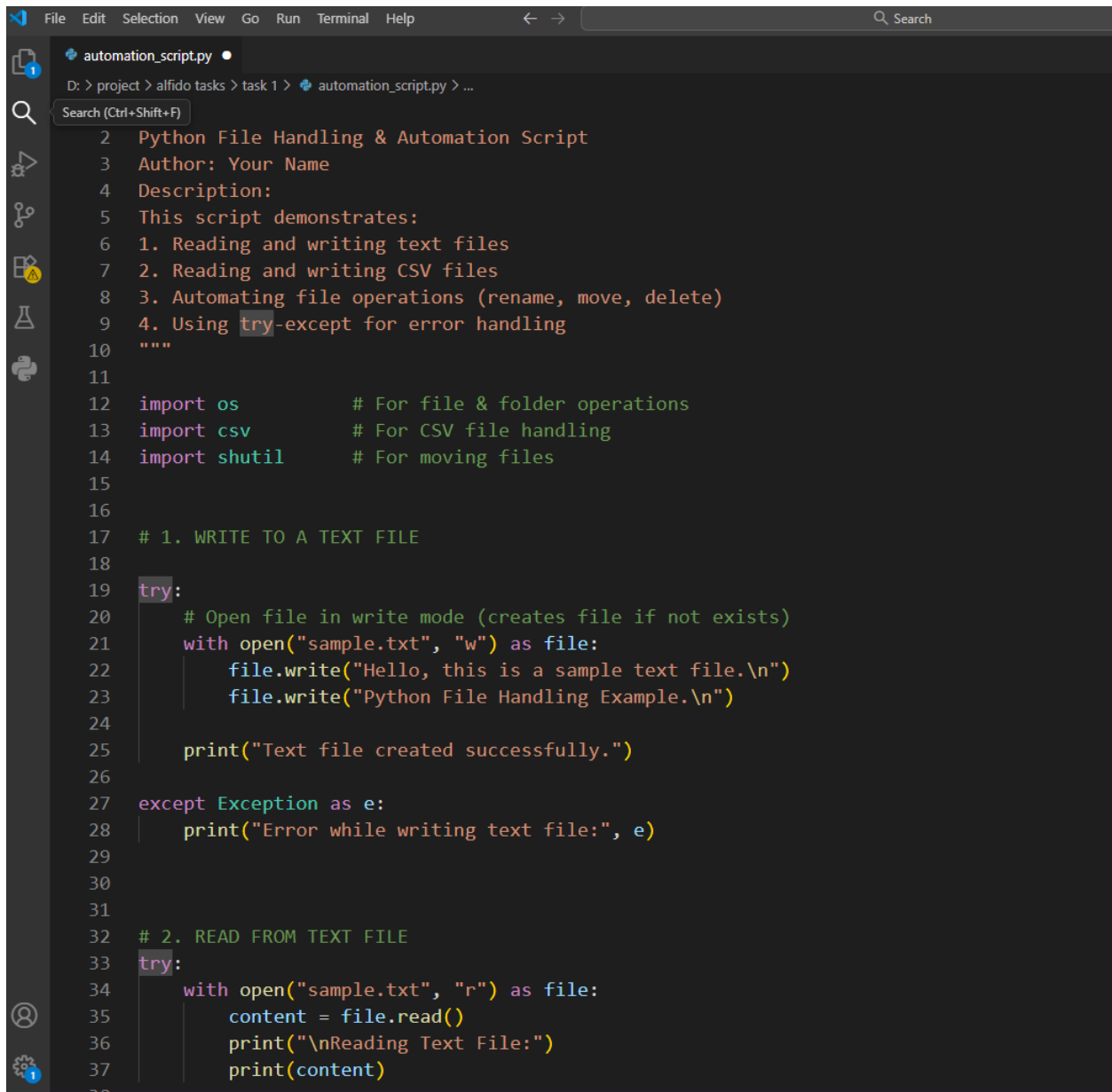
```
except FileNotFoundError:
```

```
    print("File to delete not found.")
except Exception as e:
    print("Error while deleting file:", e)

print("\nAutomation process completed.")
```

Github repository

<https://github.com/Karan6165/Alfido-Tech-Internship.git>



The image shows a code editor window with a dark theme. The title bar at the top includes a menu (File, Edit, Selection, View, Go, Run, Terminal, Help), navigation arrows, and a search bar. The left sidebar contains icons for Explorer, Search, Run and Debug, Source Control, Extensions, Testing, and a user profile. The main editor area displays a Python script named 'automation_script.py' with the following content:

```
2 Python File Handling & Automation Script
3 Author: Your Name
4 Description:
5 This script demonstrates:
6 1. Reading and writing text files
7 2. Reading and writing CSV files
8 3. Automating file operations (rename, move, delete)
9 4. Using try-except for error handling
10 """
11
12 import os          # For file & folder operations
13 import csv         # For CSV file handling
14 import shutil       # For moving files
15
16
17 # 1. WRITE TO A TEXT FILE
18
19 try:
20     # Open file in write mode (creates file if not exists)
21     with open("sample.txt", "w") as file:
22         file.write("Hello, this is a sample text file.\n")
23         file.write("Python File Handling Example.\n")
24
25     print("Text file created successfully.")
26
27 except Exception as e:
28     print("Error while writing text file:", e)
29
30
31
32 # 2. READ FROM TEXT FILE
33 try:
34     with open("sample.txt", "r") as file:
35         content = file.read()
36         print("\nReading Text File:")
37         print(content)
```

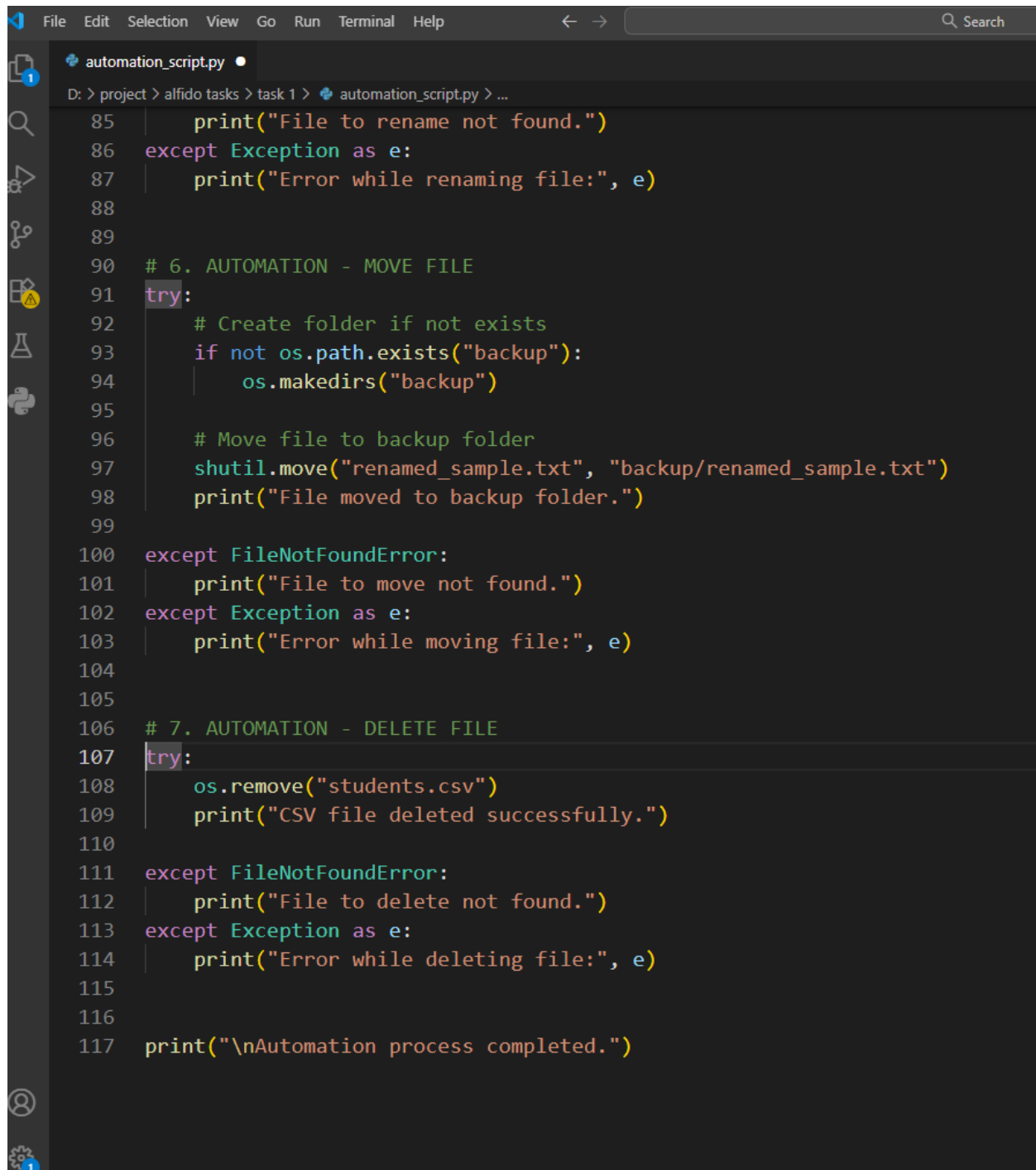
```
File Edit Selection View Go Run Terminal Help
automation_script.py
D: > project > alfidio tasks > task 1 > automation_script.py > ...

29
30
31
32 # 2. READ FROM TEXT FILE
33 try:
34     with open("sample.txt", "r") as file:
35         content = file.read()
36         print("\nReading Text File:")
37         print(content)
38
39 except FileNotFoundError:
40     print("Text file not found.")
41 except Exception as e:
42     print("Error while reading text file:", e)
43
44
45 # 3. WRITE TO CSV FILE
46 try:
47     with open("students.csv", "w", newline="") as csvfile:
48         writer = csv.writer(csvfile)
49
50         # Writing header
51         writer.writerow(["ID", "Name", "Marks"])
52
53         # Writing data rows
54         writer.writerow([1, "Karan", 85])
55         writer.writerow([2, "Rahul", 90])
56         writer.writerow([3, "Sneha", 88])
57
58     print("CSV file created successfully.")
59
60 except Exception as e:
61     print("Error while writing CSV file:", e)
62
63
64 # 4. READ FROM CSV FILE
65 try:
```

automation_script.py

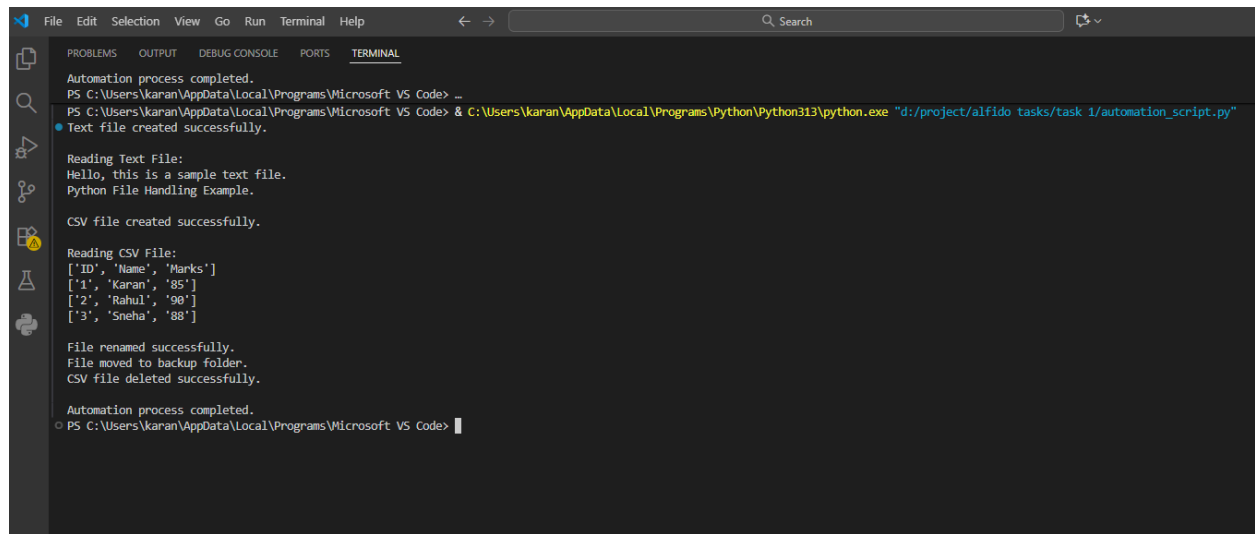
D: > project > alfredo tasks > task 1 > automation_script.py > ...

```
59
60 except Exception as e:
61     print("Error while writing CSV file:", e)
62
63
64 # 4. READ FROM CSV FILE
65 try:
66     with open("students.csv", "r") as csvfile:
67         reader = csv.reader(csvfile)
68
69         print("\nReading CSV File:")
70         for row in reader:
71             print(row)
72
73 except FileNotFoundError:
74     print("CSV file not found.")
75 except Exception as e:
76     print("Error while reading CSV file:", e)
77
78
79 # 5. AUTOMATION - RENAME FILE
80 try:
81     os.rename("sample.txt", "renamed_sample.txt")
82     print("\nFile renamed successfully.")
83
84 except FileNotFoundError:
85     print("File to rename not found.")
86 except Exception as e:
87     print("Error while renaming file:", e)
88
89
90 # 6. AUTOMATION - MOVE FILE
91 try:
92     # Create folder if not exists
93     if not os.path.exists("backup"):
94         os.makedirs("backup")
95
```

The image shows a code editor window with a dark theme. The menu bar at the top includes File, Edit, Selection, View, Go, Run, Terminal, and Help. A search bar is located on the right. The file explorer on the left shows the file 'automation_script.py'. The breadcrumb navigation indicates the path: D: > project > alfidio tasks > task 1 > automation_script.py > ...

```
85     print("File to rename not found.")
86 except Exception as e:
87     print("Error while renaming file:", e)
88
89
90 # 6. AUTOMATION - MOVE FILE
91 try:
92     # Create folder if not exists
93     if not os.path.exists("backup"):
94         os.makedirs("backup")
95
96     # Move file to backup folder
97     shutil.move("renamed_sample.txt", "backup/renamed_sample.txt")
98     print("File moved to backup folder.")
99
100 except FileNotFoundError:
101     print("File to move not found.")
102 except Exception as e:
103     print("Error while moving file:", e)
104
105
106 # 7. AUTOMATION - DELETE FILE
107 try:
108     os.remove("students.csv")
109     print("CSV file deleted successfully.")
110
111 except FileNotFoundError:
112     print("File to delete not found.")
113 except Exception as e:
114     print("Error while deleting file:", e)
115
116
117 print("\nAutomation process completed.")
```



The image shows a screenshot of a Visual Studio Code (VS Code) terminal window. The terminal is running a Python script that performs several file operations. The output of the script is displayed in the terminal, showing the successful completion of each step. The script starts by reading a text file, then creates a CSV file, reads the CSV file, renames a file, moves it to a backup folder, and finally deletes the CSV file. The terminal window has a dark theme and a sidebar on the left with various icons for file management and debugging.

```
Automation process completed.
PS C:\Users\karan\AppData\Local\Programs\Microsoft VS Code> ...
PS C:\Users\karan\AppData\Local\Programs\Microsoft VS Code> & C:\Users\karan\AppData\Local\Programs\Python\Python313\python.exe "d:/project/alfido/tasks/task 1/automation_script.py"
• Text file created successfully.

Reading Text File:
Hello, this is a sample text file.
Python File Handling Example.

CSV file created successfully.

Reading CSV File:
['ID', 'Name', 'Marks']
['1', 'Karan', '85']
['2', 'Rahul', '90']
['3', 'Sneha', '88']

File renamed successfully.
File moved to backup folder.
CSV file deleted successfully.

Automation process completed.
PS C:\Users\karan\AppData\Local\Programs\Microsoft VS Code> |
```