Handle File I/O Operations like Reading from and Writing to Files

Table of Contents

Introduction

This guide explains how to handle file I/O operations in Python, including reading from and writing to files. Understanding file handling is crucial for data persistence and manipulation in any application.

Problem Statement

Learn to create Python functions to read from and write to files. This knowledge is essential for handling data storage, configuration files, logs, and more in real-world applications.

Prerequisites

Software Requirement

- **Python 3.13.0**
  Download Python

- **Code Editor**
  A text editor or IDE like **Visual Studio Code (VS Code)** is recommended.
  Download VS Code

- **Command Line/Terminal**: For running Python scripts.

Hardware Requirement

- **Processor**: Minimum dual-core processor.
- **RAM**: 4GB or more.
- **Storage**: At least 1GB free space for Python.

Implementation Steps

Write Functions for File I/O Operations

Creating a New File

To create a new file, use the `'w'` mode with the `open()` function.

- **Create a new file**
    - Create a Python file named `creatingfile_io.py` inside your `file_operations` folder and add the following code.

```python
def create_file(filename):
    """Create a new file."""
    with open(filename, 'w') as file:
        file.write('This is a new file.')
    print(f"{filename} has been created.")

# Create a new file
create_file('example.txt')
```

- **Run the Python file**

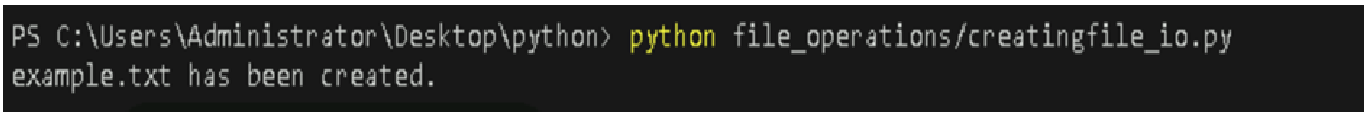
  Use the command below in your terminal to run the Python file:

```
python file_operations/creatingfile_io.py
```

Alternatively, you can use:

```
cd file_operations
python creatingfile_io.py
```

Output:



```
PS C:\Users\Administrator\Desktop\python> python file_operations/creatingfile_io.py
example.txt has been created.
```

This function creates a new file and writes a line of text to it.

Writing to a File

- **Create a new file**
    - Create a Python file named `writingfile_io.py` inside your `file_operations` folder and add the following code.

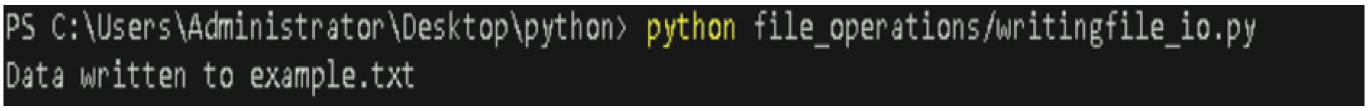
```
def write_to_file(filename, data):
    """Write data to a file."""
    with open(filename, 'w') as file:
        file.write(data)
        print(f'Data written to {filename}')

write_to_file('example.txt', 'Hello, this is a test file!')
```

- **Run the Python file**

  Use the command below in your terminal to run the Python file:

```
python file_operations/writingfile_io.py
```

Output:



```
PS C:\Users\Administrator\Desktop\python> python file_operations/writingfile_io.py
Data written to example.txt
```

This function writes the specified data to the given file.

Reading from a File

- **Create a new file**
    - Create a Python file named `readingfile_io.py` inside your `file_operations` folder and add the following code.

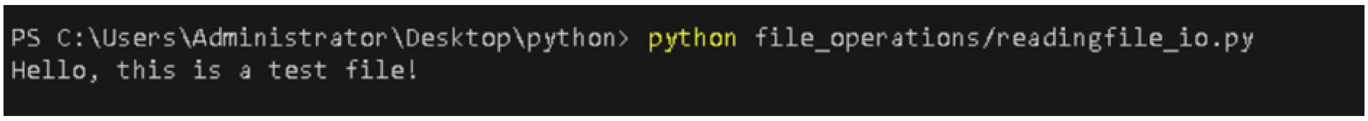
```
def read_from_file(filename):
    """Read data from a file."""
    with open(filename, 'r') as file:
        return file.read()

data = read_from_file('example.txt')
print(data)
```

- **Run the Python file**

Use the command below in your terminal to run the Python file:

```
python file_operations/readingfile_io.py
```

Output:



```
PS C:\Users\Administrator\Desktop\python> python file_operations/readingfile_io.py
Hello, this is a test file!
```

This function reads the entire content of the specified file.*

Appending to a File

- **Create a new file**
  - Create a Python file named `appending_io.py` inside your `file_operations` folder and add the following code.

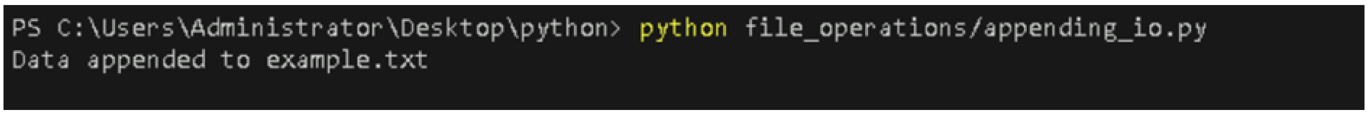
```python
def append_to_file(filename, data):
    """Append data to a file."""
    with open(filename, 'a') as file:
        file.write(data)
        print(f"Data appended to {filename}")

append_to_file('example.txt', '\nThis line has been appended.')
```

- **Run the Python file**

  Use the command below in your terminal to run the Python file:

```
python file_operations/appending_io.py
```

Output:



```
PS C:\Users\Administrator\Desktop\python> python file_operations/appending_io.py
Data appended to example.txt
```

This function appends data to the end of the specified file.

Reading Specific Lines from a File

- **Create a new file**
  - Create a Python file named `specific_line.py` inside your `file_operations` folder and add the following code.

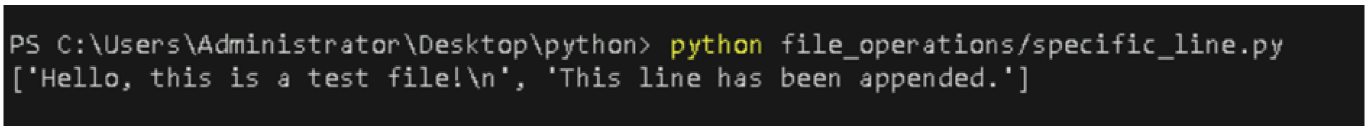
```
def read_specific_lines(filename, start_line, end_line):
    """Read specific lines from a file."""
    with open(filename, 'r') as file:
        lines = file.readlines()
        return lines[start_line:end_line]

specific_lines = read_specific_lines('example.txt', 0, 2)
print(specific_lines)
```

- **Run the Python file**

  Use the command below in your terminal to run the Python file:

```
python file_operations/specific_line.py
```

Output:



```
PS C:\Users\Administrator\Desktop\python> python file_operations/specific_line.py
['Hello, this is a test file!\n', 'This line has been appended.']
```

This function reads and returns specific lines from the file based on the given line numbers.

Checking if a File Exists

- **Create a new file**
  - Create a Python file named `checking_file.py` inside your `file_operations` folder and add the following code.

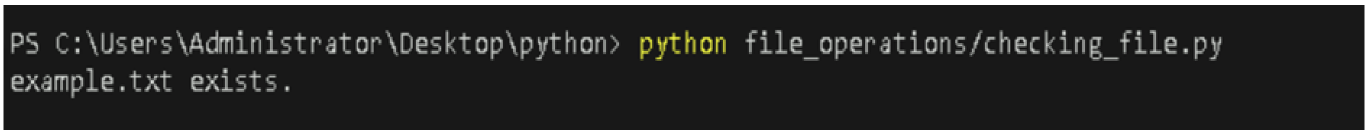
```
import os

def check_file_exists(filename):
    """Check if a file exists."""
    if os.path.isfile(filename):
        print(f"{filename} exists.")
    else:
        print(f"{filename} does not exist.")

check_file_exists('example.txt')
```

- **Run the Python file**

  Use the command below in your terminal to run the Python file:

```
python file_operations/checking_file.py
```

Output:



```
PS C:\Users\Administrator\Desktop\python> python file_operations/checking_file.py
example.txt exists.
```

This function checks if the specified file exists.

Copying a File

- **Create a new file**
  - Create a Python file named `copying_file.py` inside your `file_operations` folder and add the following code.

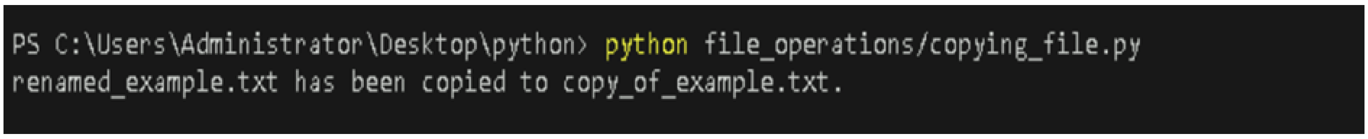
```python
import shutil

def copy_file(source, destination):
    """Copy a file from source to destination."""
    shutil.copy(source, destination)
    print(f"{source} has been copied to {destination}.")

copy_file('renamed_example.txt', 'copy_of_example.txt')
```

- **Run the Python file**

  Use the command below in your terminal to run the Python file:

```
python file_operations/copying_file.py
```

Output:



```
PS C:\Users\Administrator\Desktop\python> python file_operations/copying_file.py
renamed_example.txt has been copied to copy_of_example.txt.
```

This function copies a file from one location to another.

Renaming a File

- **Create a new file**
  - Create a Python file named `renaming_file.py` inside your `file_operations` folder and add the following code.

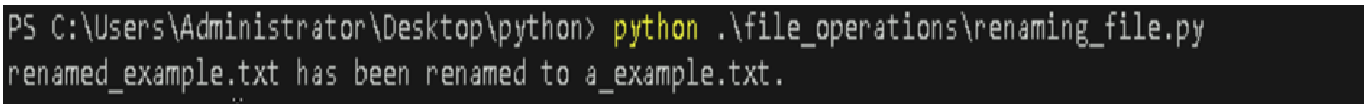
```
import os

def rename_file(old_name, new_name):
    """Rename a file."""
    os.rename(old_name, new_name)
    print(f"{old_name} has been renamed to {new_name}.")

rename_file('renamed_example.txt', 'a_example.txt')
```

- **Run the Python file**

  Use the command below in your terminal to run the Python file:

```
python file_operations/renaming_file.py
```

Output:



```
PS C:\Users\Administrator\Desktop\python> python .\file_operations\renaming_file.py
renamed_example.txt has been renamed to a_example.txt.
```

This function renames the specified file.

Deleting a File

- **Create a new file**
  - Create a Python file named `deleting_file.py` inside your `file_operations` folder and add the following code.

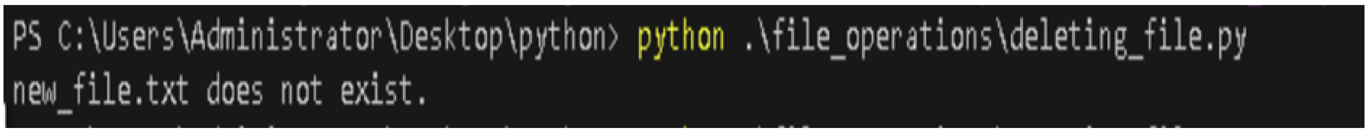
```
import os
def delete_file(filename):
    """Delete a file if it exists."""
    if os.path.isfile(filename):
        os.remove(filename)
        print(f"{filename} has been deleted.")
    else:
        print(f"{filename} does not exist.")

delete_file('new_file.txt')
```

- **Run the Python file**

  Use the command below in your terminal to run the Python file:

```
python file_operations/deleting_file.py
```

Output:



```
PS C:\Users\Administrator\Desktop\python> python .\file_operations\deleting_file.py
new_file.txt does not exist.
```

This function deletes the specified file if it exists.

Reading CSV Files

- **Create a new file**
  - Create a Python file named `reading_csv.py` inside your `file_operations` folder and add the following code.

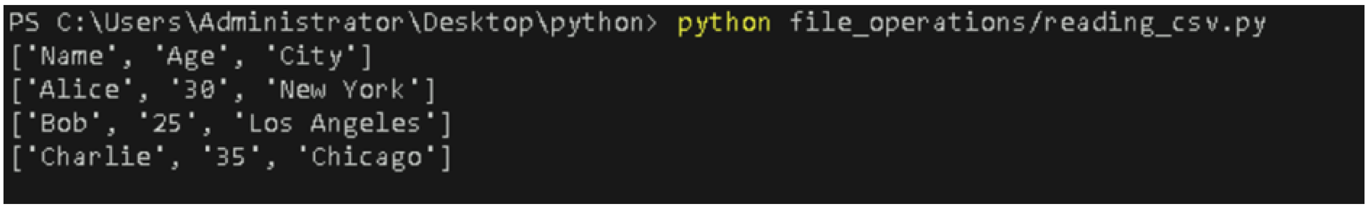
```
import csv

def read_csv_file(filename):
    """Read a CSV file."""
    with open(filename, 'r') as csvfile:
        reader = csv.reader(csvfile)
        for row in reader:
            print(row)

read_csv_file('sample.csv')
```

- **Run the Python file**

  Use the command below in your terminal to run the Python file:

```
python file_operations/reading_csv.py
```

Output:



```
PS C:\Users\Administrator\Desktop\python> python file_operations/reading_csv.py
['Name', 'Age', 'City']
['Alice', '30', 'New York']
['Bob', '25', 'Los Angeles']
['Charlie', '35', 'Chicago']
```

This function reads data from a CSV file.

Writing CSV Files

- **Create a new file**
  - Create a Python file named `writing_csv.py` inside your `file_operations` folder and add the following code.

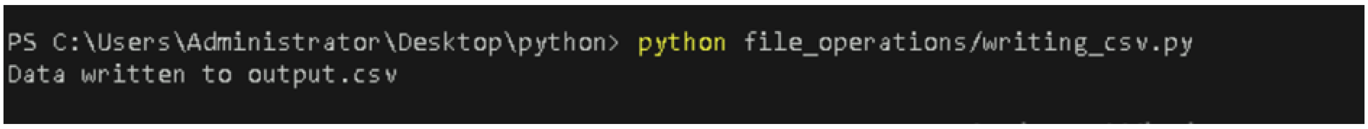
```python
import csv

def write_csv_file(filename, data):
    """Write data to a CSV file."""
    with open(filename, 'w', newline='') as csvfile:
        writer = csv.writer(csvfile)
        writer.writerows(data)
        print(f'Data written to {filename}')

data_to_write = [['Name', 'Age'], ['Alice', 30], ['Bob', 25]]
write_csv_file('output.csv', data_to_write)
```

- **Run the Python file**

  Use the command below in your terminal to run the Python file:

```
python file_operations/writing_csv.py
```

Output:



```
PS C:\Users\Administrator\Desktop\python> python file_operations/writing_csv.py
Data written to output.csv
```

This function writes data to a CSV file.

References

- Python Official Documentation
- W3Schools - Python File Handling
- Python File I/O
- Working with Files in Python
