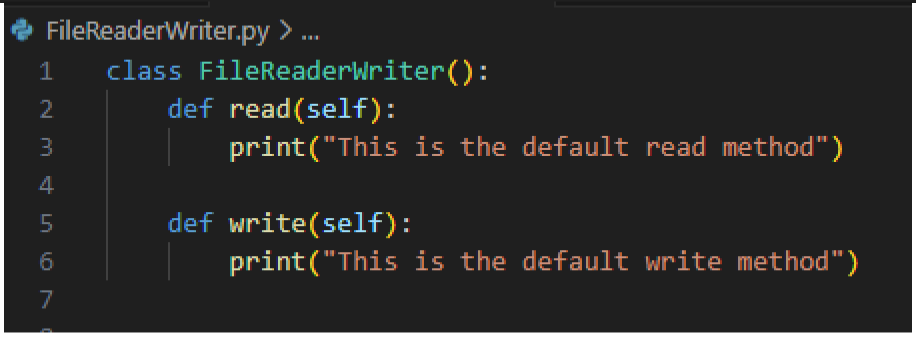
Laboratory Activity # 3	
Polymorphism	
Course Code: CPE009B	**Program:** BSCPE
Course Title: Object-Oriented Programming	**Date Performed:** September 30, 2024
Section: CPE21S4	**Date Submitted:** September 30, 2024
Name: Bona, Andrei Nycole So	**Instructor:** Prof. Maria Rizette Sayo
6. Supplementary Activity:	

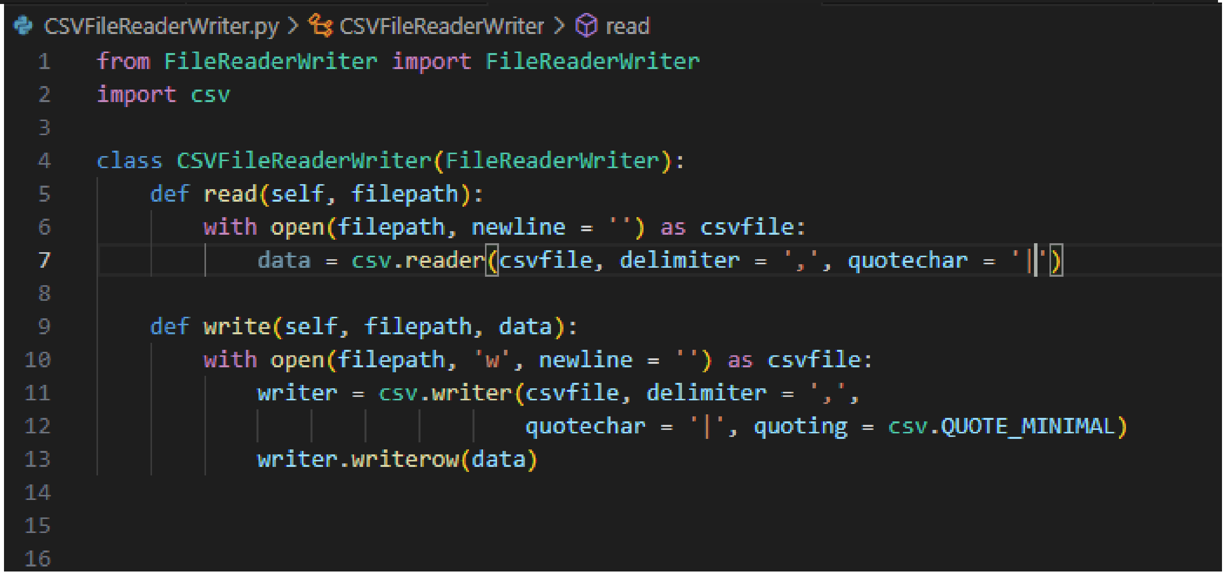
FileReaderWriter



FileReaderWriter.py > ...

```python
class FileReaderWriter():
    def read(self):
        print("This is the default read method")

    def write(self):
        print("This is the default write method")

```

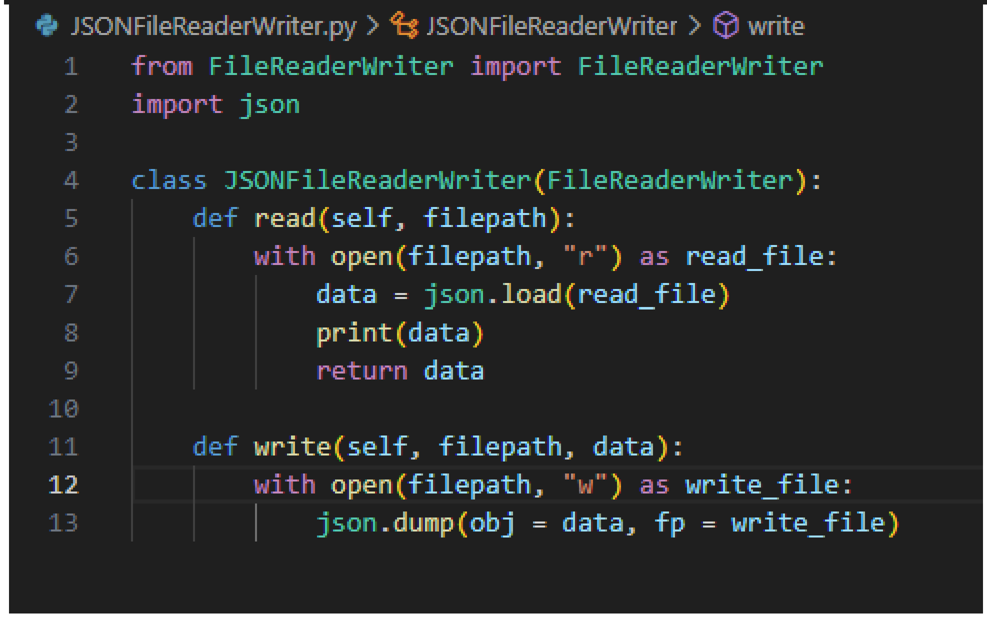
CSVFileReaderWriter



CSVFileReaderWriter.py > CSVFileReaderWriter > read

```python
from FileReaderWriter import FileReaderWriter
import csv

class CSVFileReaderWriter(FileReaderWriter):
    def read(self, filepath):
        with open(filepath, newline = '') as csvfile:
            data = csv.reader(csvfile, delimiter = ',', quotechar = '|')

    def write(self, filepath, data):
        with open(filepath, 'w', newline = '') as csvfile:
            writer = csv.writer(csvfile, delimiter = ',',
                                quotechar = '|', quoting = csv.QUOTE_MINIMAL)
            writer.writerow(data)



```

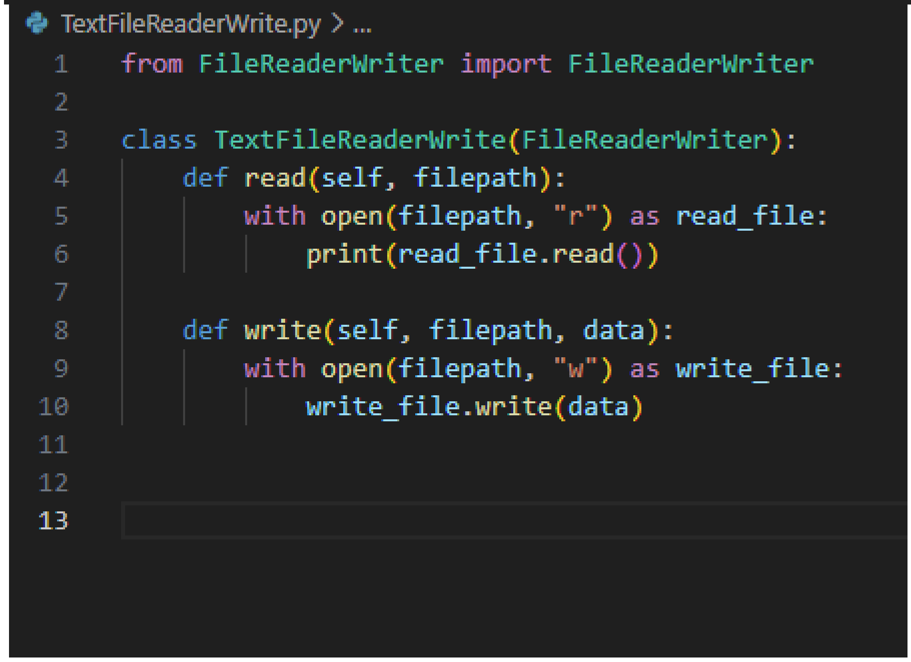
JSONFileReaderWriter



JSONFileReaderWriter.py > JSONFileReaderWriter > write

```python
from FileReaderWriter import FileReaderWriter
import json

class JSONFileReaderWriter(FileReaderWriter):
    def read(self, filepath):
        with open(filepath, "r") as read_file:
            data = json.load(read_file)
            print(data)
            return data

    def write(self, filepath, data):
        with open(filepath, "w") as write_file:
            json.dump(obj = data, fp = write_file)
```

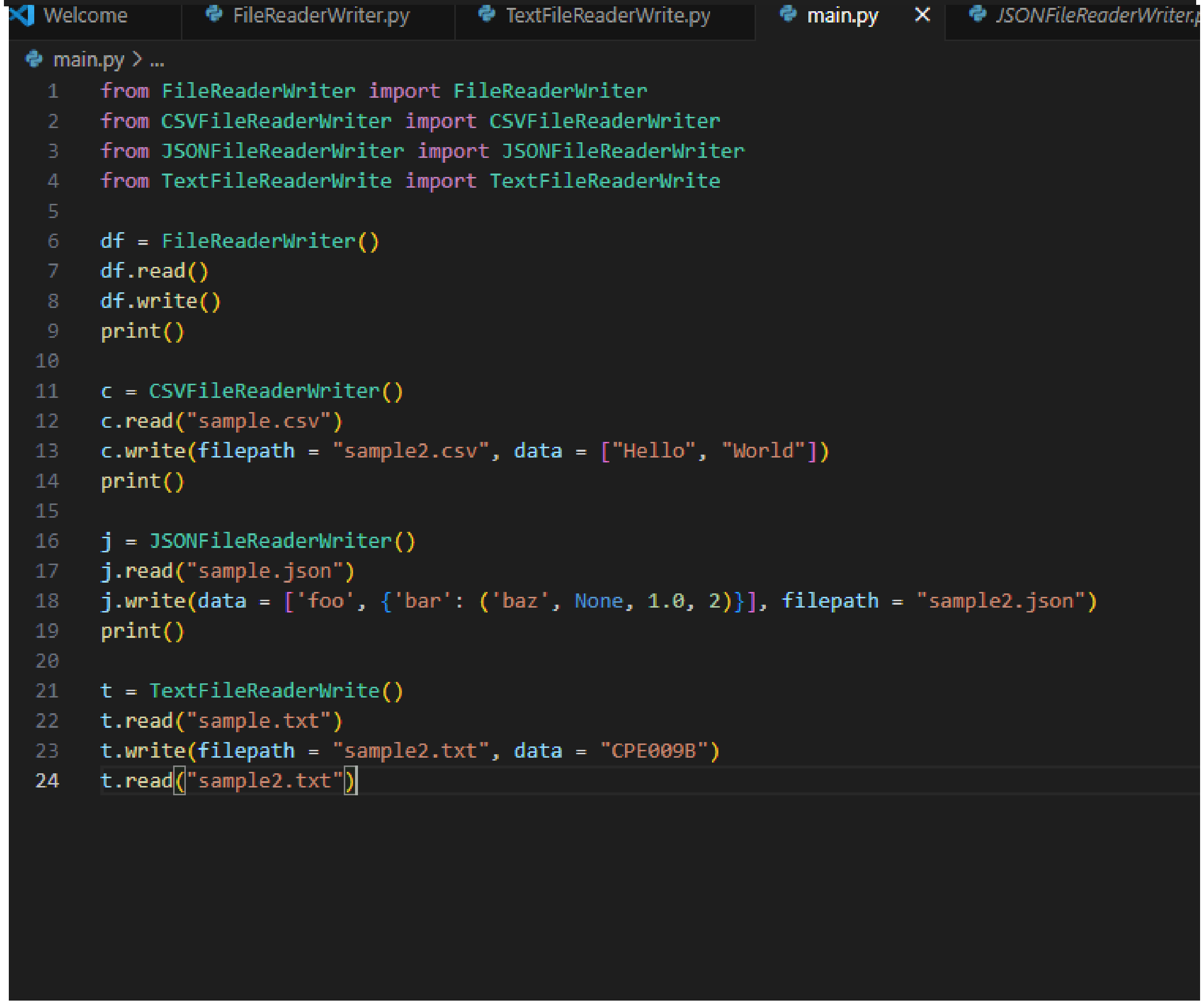
TextFileReaderWrite



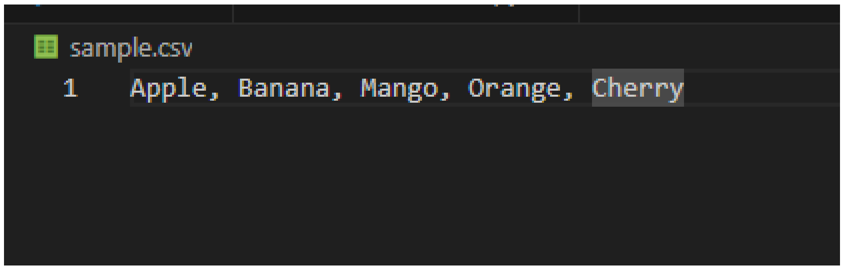
TextFileReaderWrite.py > ...

```python
from FileReaderWriter import FileReaderWriter

class TextFileReaderWrite(FileReaderWriter):
    def read(self, filepath):
        with open(filepath, "r") as read_file:
            print(read_file.read())

    def write(self, filepath, data):
        with open(filepath, "w") as write_file:
            write_file.write(data)


```

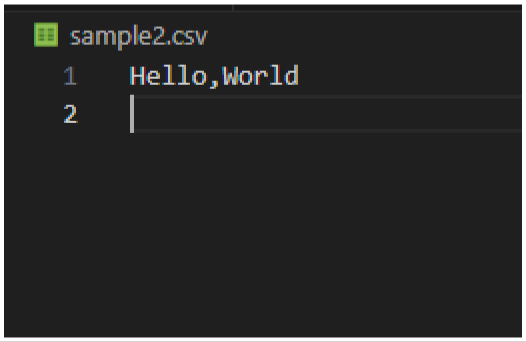
Main



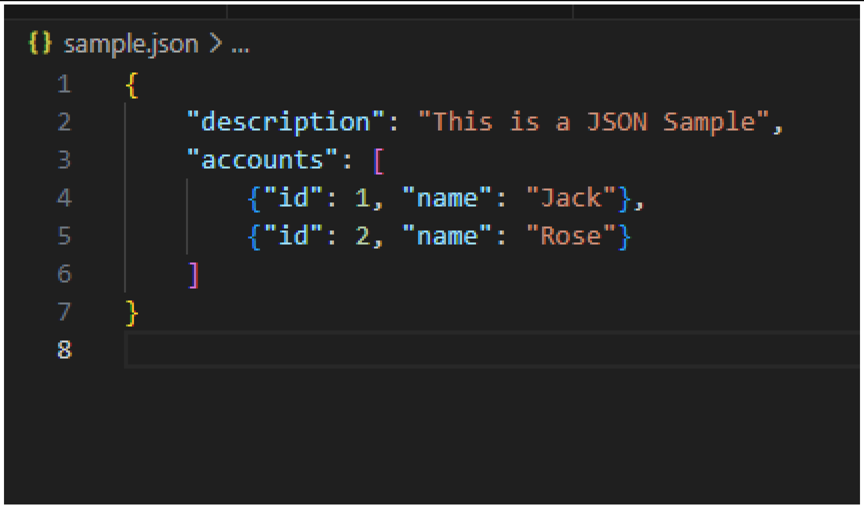
```python
from FileReaderWriter import FileReaderWriter
from CSVFileReaderWriter import CSVFileReaderWriter
from JSONFileReaderWriter import JSONFileReaderWriter
from TextFileReaderWrite import TextFileReaderWrite

df = FileReaderWriter()
df.read()
df.write()
print()

c = CSVFileReaderWriter()
c.read("sample.csv")
c.write(filepath = "sample2.csv", data = ["Hello", "World"])
print()

j = JSONFileReaderWriter()
j.read("sample.json")
j.write(data = ['foo', {'bar': ('baz', None, 1.0, 2)}], filepath = "sample2.json")
print()

t = TextFileReaderWrite()
t.read("sample.txt")
t.write(filepath = "sample2.txt", data = "CPE009B")
t.read("sample2.txt")
```


sample.csv

```
Apple, Banana, Mango, Orange, Cherry
```

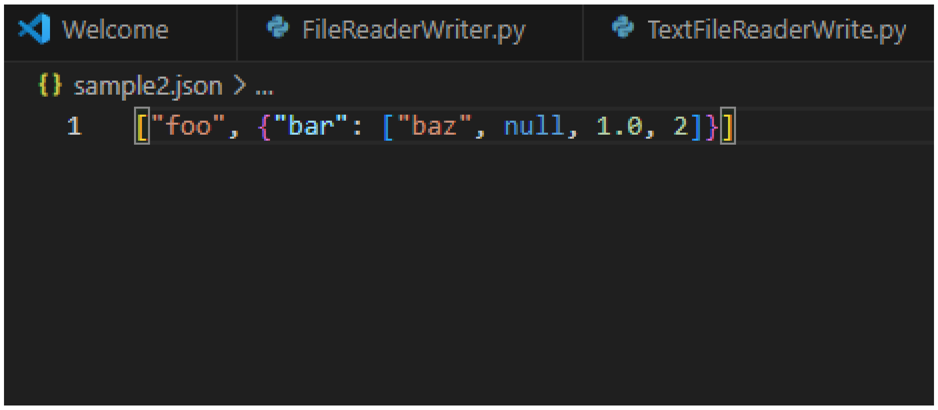

sample2.csv

```
Hello,World

```



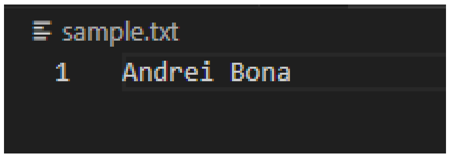
sample.json > ...

```json
{
    "description": "This is a JSON Sample",
    "accounts": [
        {"id": 1, "name": "Jack"},
        {"id": 2, "name": "Rose"}
    ]
}

```



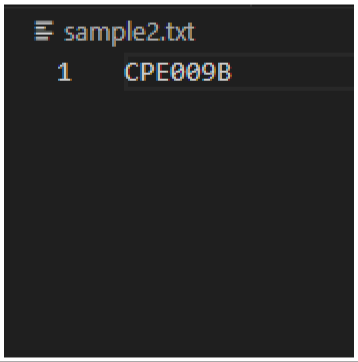
sample2.json > ...

```json
["foo", {"bar": ["baz", null, 1.0, 2]}]
```



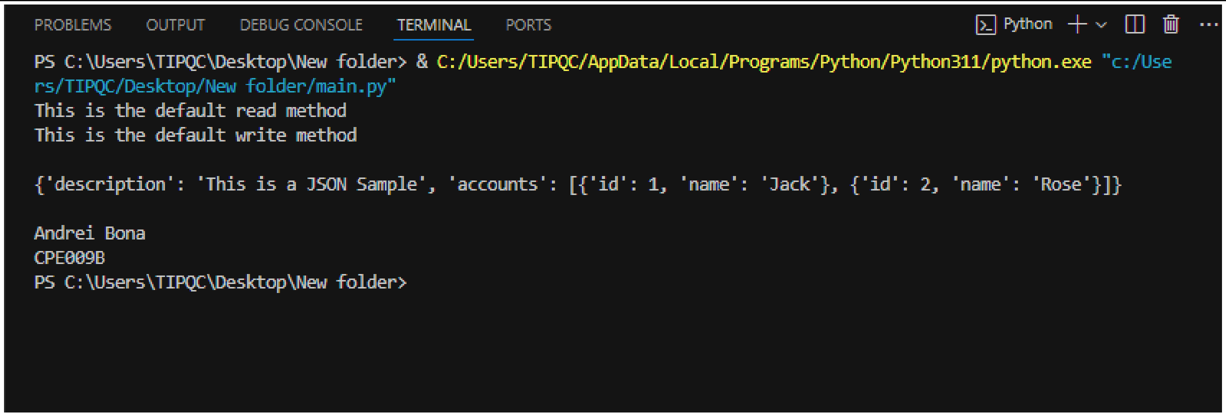
sample.txt

```
Andrei Bona
```

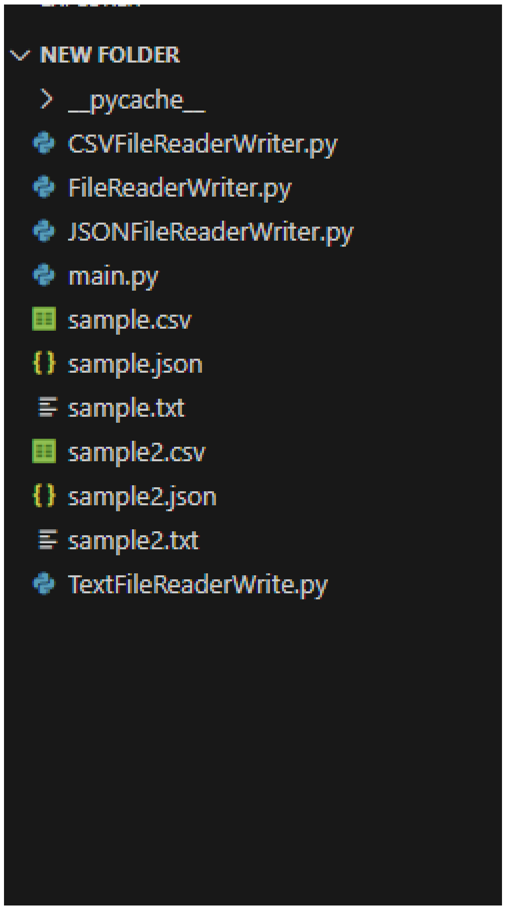


sample2.txt

```
CPE009B
```



```
PS C:\Users\TIPQC\Desktop\New folder> & C:/Users/TIPQC/AppData/Local/Programs/Python/Python311/python.exe "c:/Use
rs/TIPQC/Desktop/New folder/main.py"
This is the default read method
This is the default write method

{'description': 'This is a JSON Sample', 'accounts': [{'id': 1, 'name': 'Jack'}, {'id': 2, 'name': 'Rose'}]}

Andrei Bona
CPE009B
PS C:\Users\TIPQC\Desktop\New folder>
```



```
NEW FOLDER
  > __pycache__
  CSVFileReaderWriter.py
  FileReaderWriter.py
  JSONFileReaderWriter.py
  main.py
  sample.csv
  sample.json
  sample.txt
  sample2.csv
  sample2.json
  sample2.txt
  TextFileReaderWrite.py
```

Questions

1. Why is Polymorphism important?

Polymorphism makes various classes share a common superclass' features, thus, making the code more flexible, reusable, and less complex.

2. Explain the advantages and disadvantages of using applying Polymorphism in an Object-Oriented Program.

Polymorphism has several benefits and drawbacks. Its advantages include code reusability, allowing different classes to share the same code, flexibility in adding new classes without changes to existing ones, and easier maintenance, as changes in one class don't affect others. However, it can be complex for beginners, may introduce performance issues due to dynamic method calls, and can make debugging harder.

3. What maybe the advantage and disadvantage of the program we wrote to read and write csv and json files?

CSV and JSON file handling comes with pros and cons. The advantages are flexibility in supporting various data formats, simplicity since both are easy to read and edit, and their widespread use, which ensures compatibility with many tools. On the downside, there can be data integrity issues, performance slowdowns with large files, and limitations like CSV's struggle with hierarchical data and JSON's lack of schema validation.

4. What maybe considered if Polymorphism is to be implemented in an Object-Oriented Program?

When implementing polymorphism, keep these points in mind: define clear interfaces or abstract classes for shared behavior, use design patterns to organize your code, ensure type safety to avoid errors, and maintain good documentation to explain class interactions.

5. How do you think Polymorphism is used in an actual programs that we use today?

Polymorphism is widely used in real applications. For instance, in GUI frameworks, UI components respond to events uniformly. In APIs, different classes can work under a common interface, and in data processing libraries, functions can handle various data types seamlessly.

7. Conclusion:

Polymorphism is essential in object-oriented programming, offering flexibility and reusability while introducing some complexity. When applied effectively, it leads to more adaptable and maintainable software, benefiting developers and users alike.

8. Assessment Rubric: