Laboratory Activity No. 3	
Polymorphism	
Course Code: CPE009	**Program:** BSCPE
Course Title: Object-Oriented Programming	**Date Performed:** 09/30/24
Section: CpE21S4	**Date Submitted:** 09/30/24
Name: Edson Ray E. San Juan	**Instructor:** Prof. Maria Rizette Sayo

1. Objective(s):

This activity aims to familiarize students with the concepts of Polymorphism in Object-Oriented Programming

2. Intended Learning Outcomes (ILOs):

The students should be able to:

2.1 Identify the use of Polymorphism in Object-Oriented Programming

2.2 Implement an Object-Oriented Program that applies Polymorphism

3. Discussion:

Polymorphism is a core principle of Object-Oriented that is also called “method overriding”. Simply stated the principles says
that a method can be redefined to have a different behavior in different derived classees.

For an example, consider a **base file reader/writer** class then three derived classes **Text file reader/writer**, **CSV file reader/ writer**, and **JSON file reader/writer**. The base file reader/writer class has the methods: **read**(filepath=””) , **write**(filepath=””). The three derived classes (classes that would inherit from the base class) should have behave differently when their read, write methods are invoked.

CSV stands for **Comma Separated Values** while **JSON** stands for **Javascript Server Object Notation**. These are the standard file formats and structures used by applications and systems to transfer/exchange data between their systems. For example, you may visit this online api http://dummy.restapiexample.com/api/v1/employees (note that the data is fake) but this url provides data that another system can consume and use in their system.

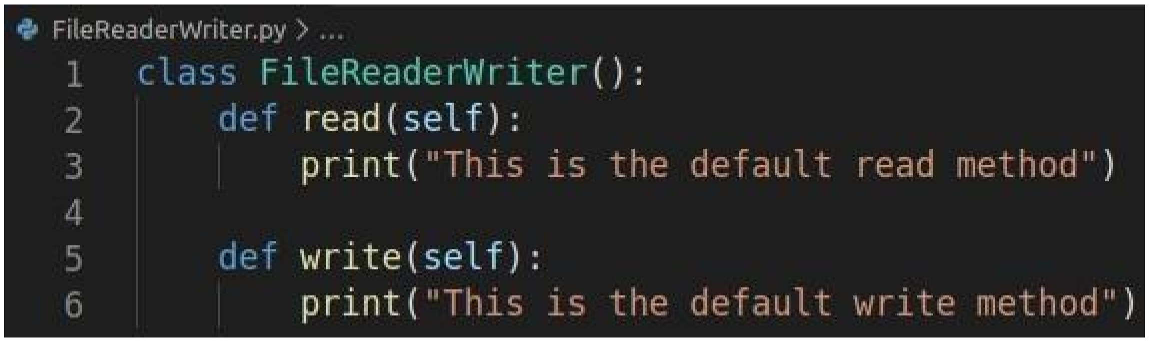
4. Materials and Equipment:

Desktop Computer with Anaconda
Python Windows Operating System

5. Procedure:

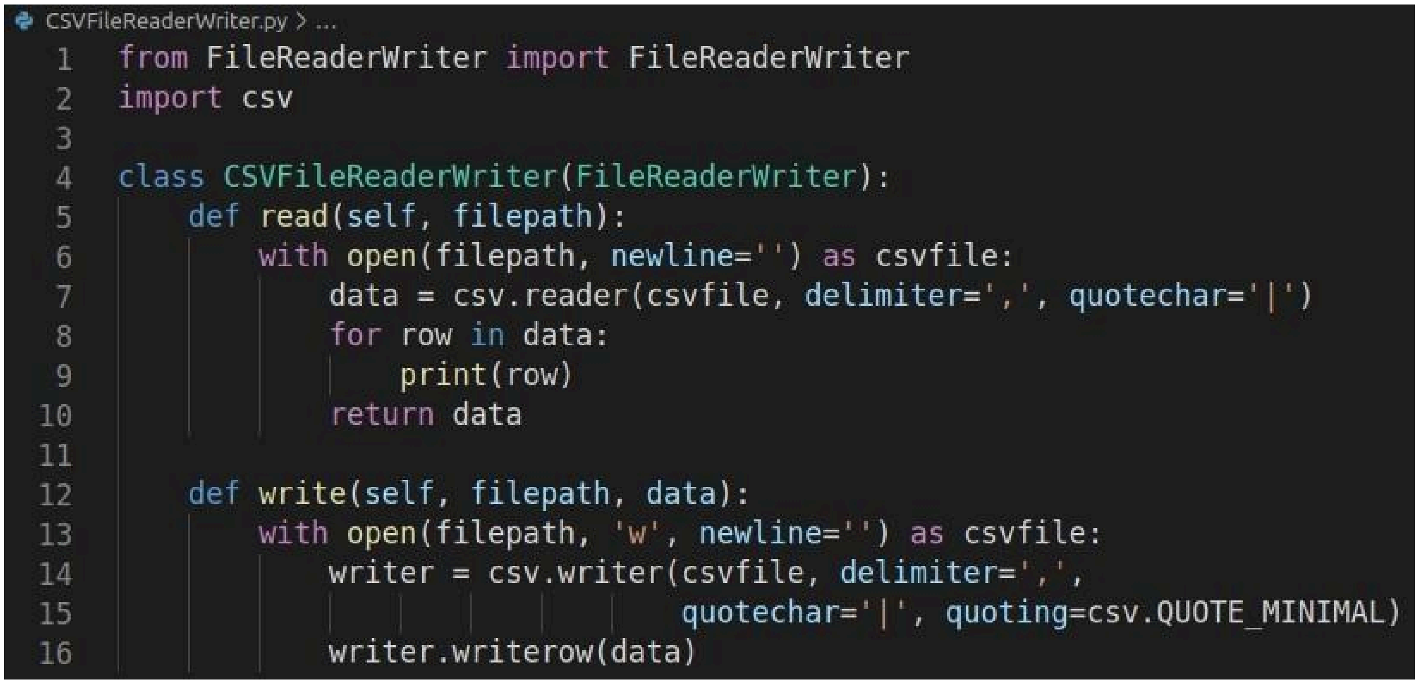
Creating the Classes

1. Create a folder named oopfa1<lastname>_lab8
2. Open your IDE in that folder.
3. Create the base FileReaderWriter .py file and Class using the code below:



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

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

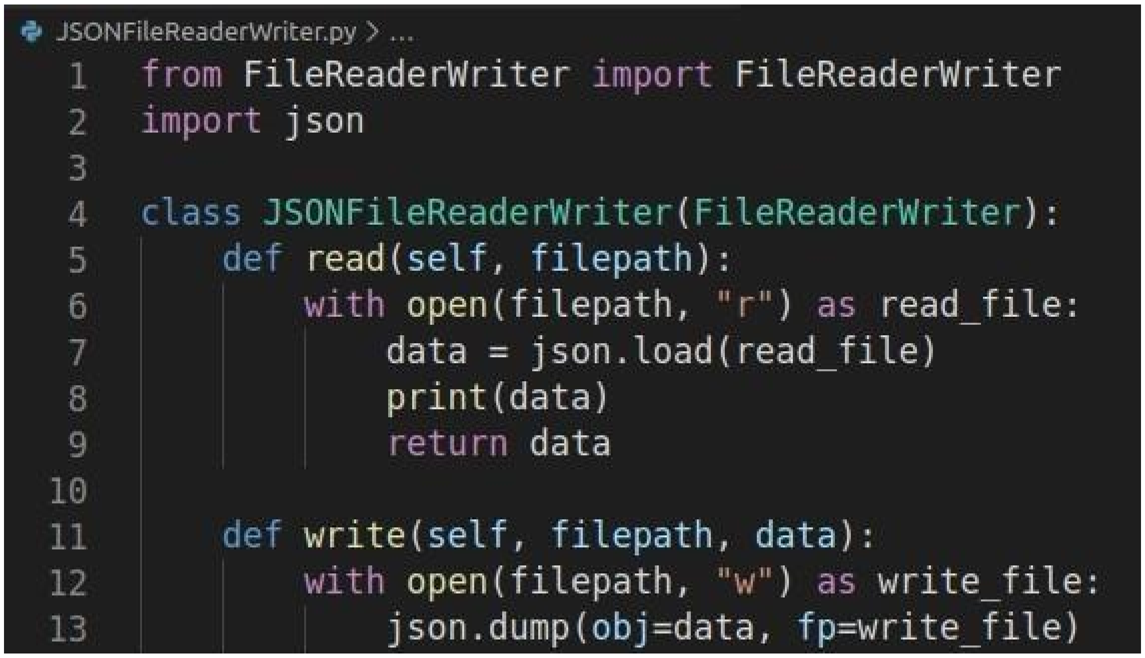
4. Create the CSVFileReaderWriter .py and Class using the code below:



CSVFileReaderWriter.py > ...

```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='|')
            for row in data:
                print(row)
            return data

    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)
```

5. Create the JSONFileReaderWriter Class using the code below

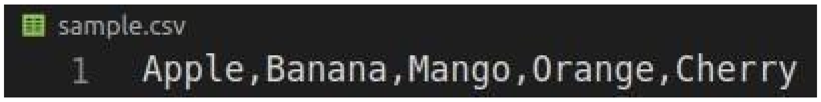


JSONFileReaderWriter.py > ...

```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)
```

Testing and Observing Polymorphism

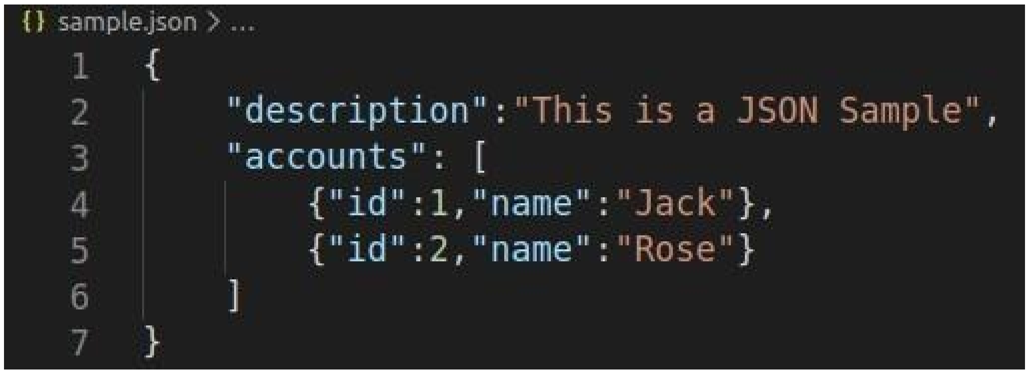
1. Create a .csv file named sample.csv with the following content. (you may use the IDE or plain notepad)



sample.csv

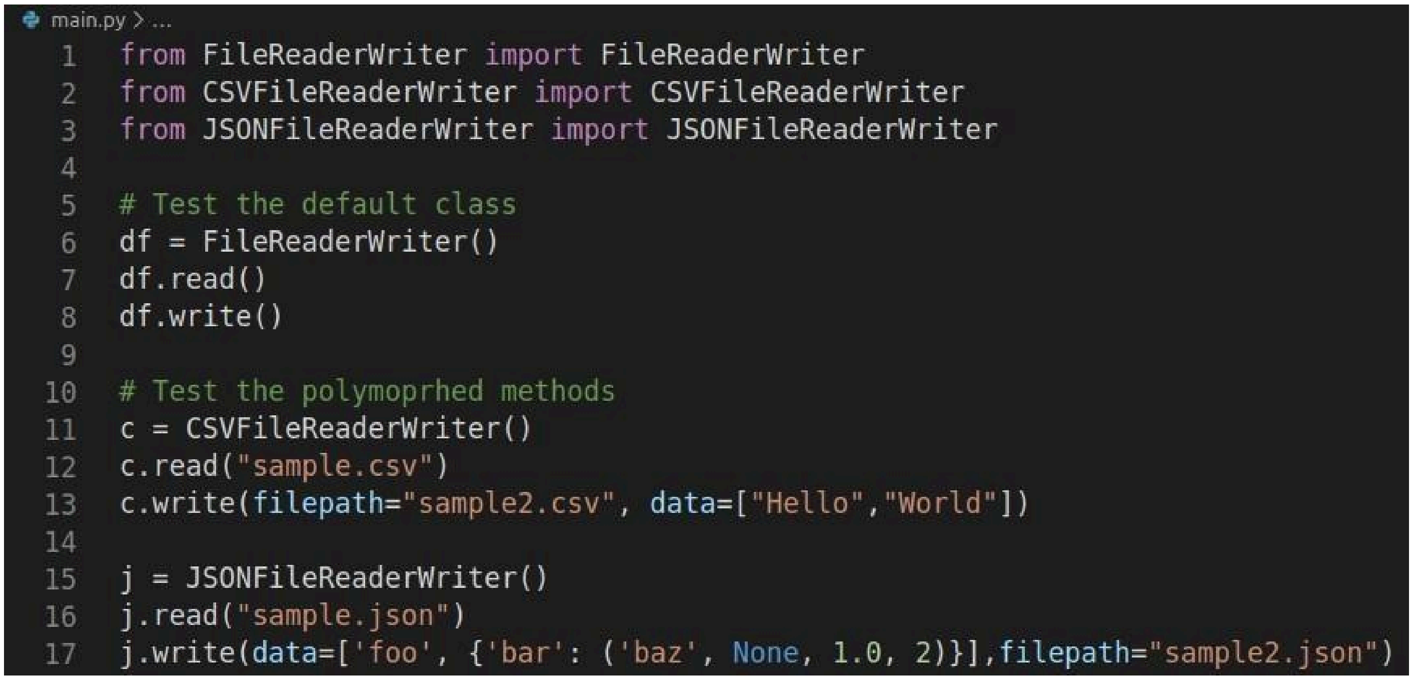
```
Apple,Banana,Mango,Orange,Cherry
```

2. Create a .json file named sample.json with the following content. (you may use the IDE or plain notepad)



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

3. Create the main.py that will test the functionality of the classes.



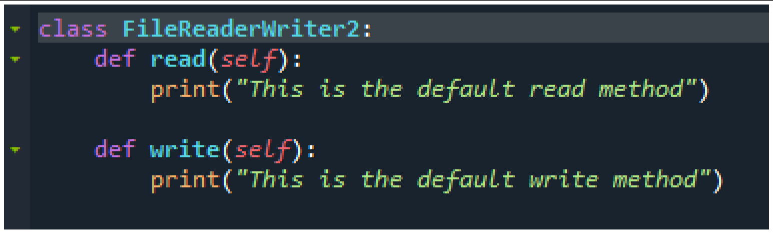
```python
from FileReaderWriter import FileReaderWriter
from CSVFileReaderWriter import CSVFileReaderWriter
from JSONFileReaderWriter import JSONFileReaderWriter

# Test the default class
df = FileReaderWriter()
df.read()
df.write()

# Test the polymoprhed methods
c = CSVFileReaderWriter()
c.read("sample.csv")
c.write(filepath="sample2.csv", data=["Hello","World"])

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

4. Run the program and observe the output carefully the values in sample2.csv and sample2.json.

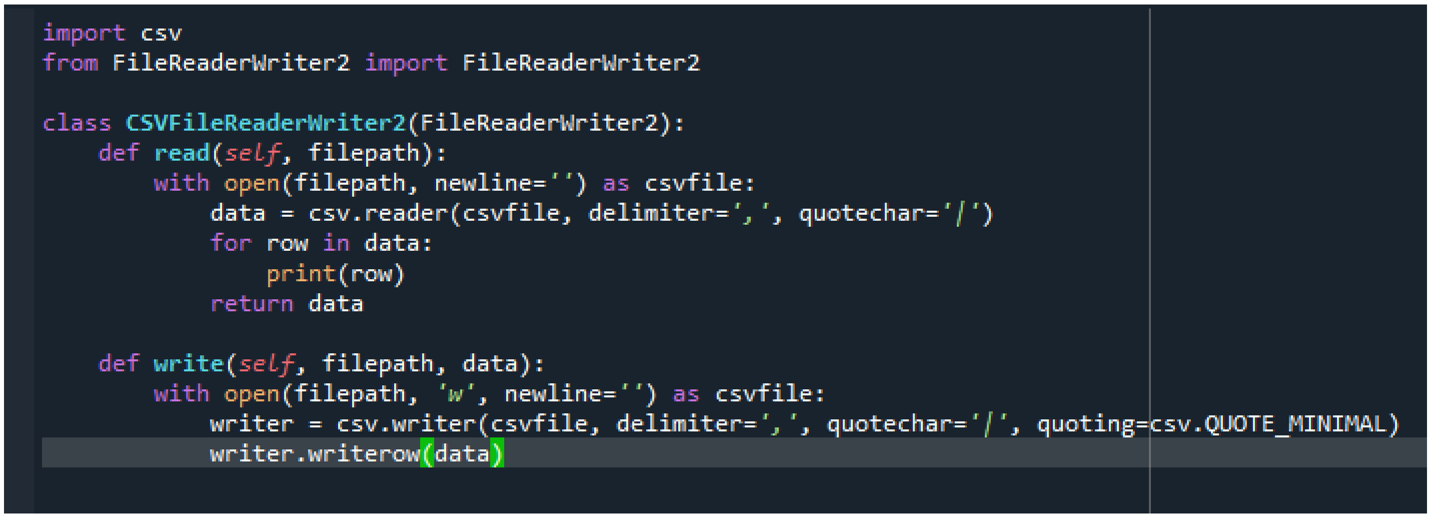
6. Supplementary Activity:

Task

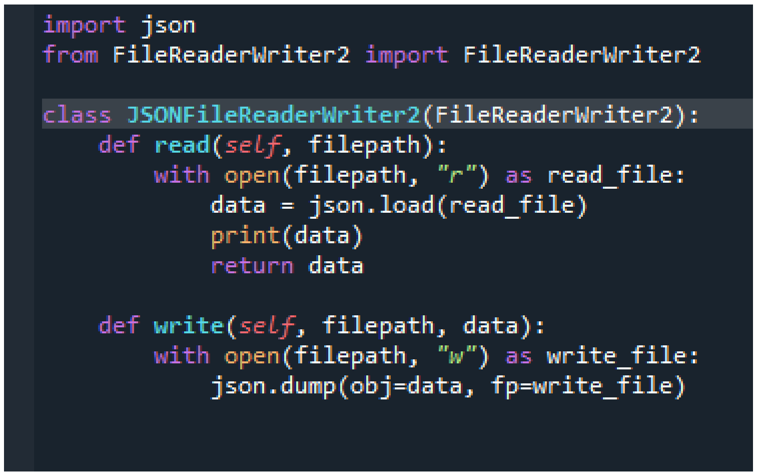
Create a simple TextFileReaderWriter .py file and Class that will be able to **read** from and **write** (override) to a text file. The read and write method should be overriden according to the requirement of Text File Reading and Writing as performed in Laboratory Activity 5.



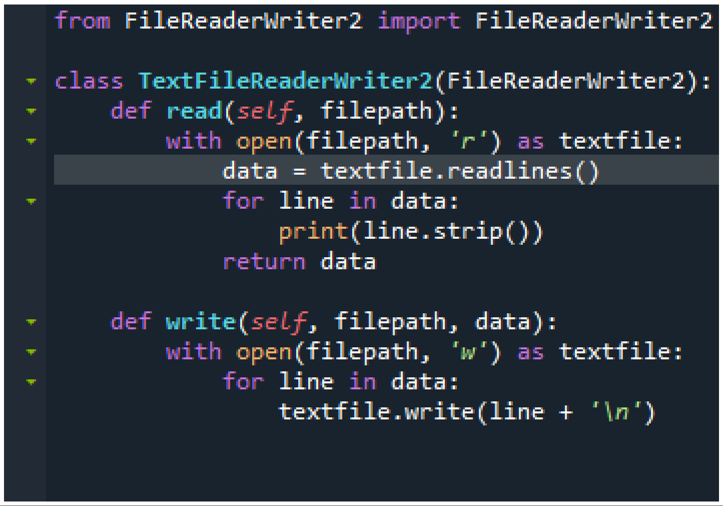
```python
class FileReaderWriter2:
    def read(self):
        print("This is the default read method")

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



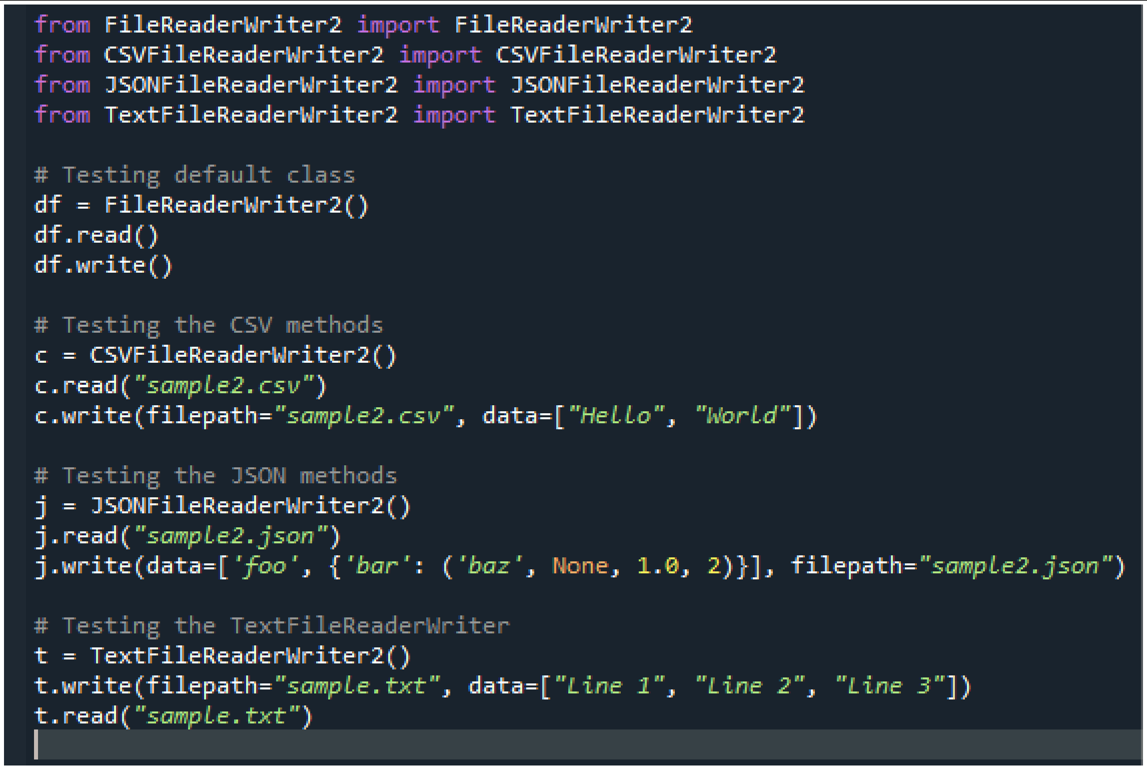
```python
import csv
from FileReaderWriter2 import FileReaderWriter2

class CSVFileReaderWriter2(FileReaderWriter2):
    def read(self, filepath):
        with open(filepath, newline='') as csvfile:
            data = csv.reader(csvfile, delimiter=',', quotechar='|')
            for row in data:
                print(row)
            return data

    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)
```



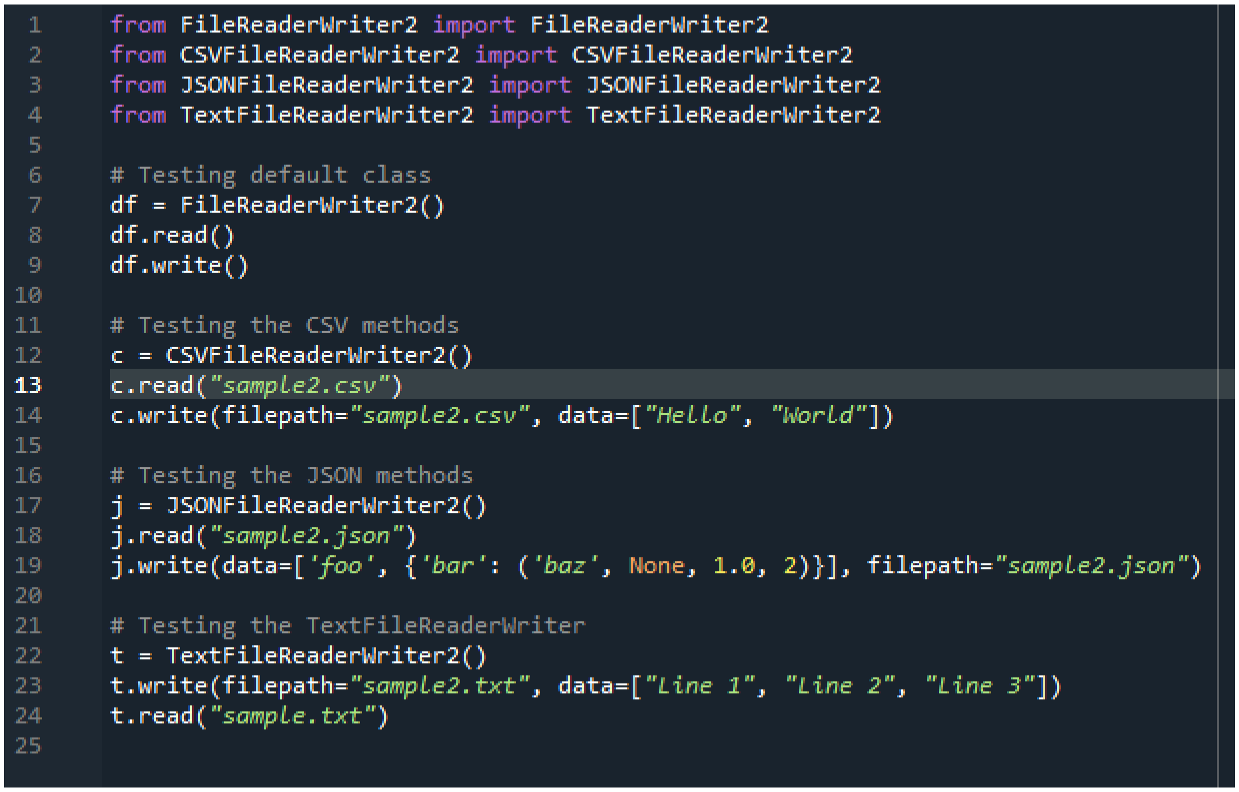
```python
import json
from FileReaderWriter2 import FileReaderWriter2

class JSONFileReaderWriter2(FileReaderWriter2):
    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)
```



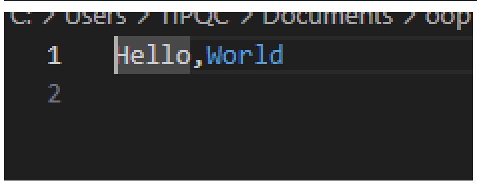
```python
from FileReaderWriter2 import FileReaderWriter2

class TextFileReaderWriter2(FileReaderWriter2):
    def read(self, filepath):
        with open(filepath, 'r') as textfile:
            data = textfile.readlines()
            for line in data:
                print(line.strip())
            return data

    def write(self, filepath, data):
        with open(filepath, 'w') as textfile:
            for line in data:
                textfile.write(line + '\n')
```



```python
from FileReaderWriter2 import FileReaderWriter2
from CSVFileReaderWriter2 import CSVFileReaderWriter2
from JSONFileReaderWriter2 import JSONFileReaderWriter2
from TextFileReaderWriter2 import TextFileReaderWriter2

# Testing default class
df = FileReaderWriter2()
df.read()
df.write()

# Testing the CSV methods
c = CSVFileReaderWriter2()
c.read("sample2.csv")
c.write(filepath="sample2.csv", data=["Hello", "World"])

# Testing the JSON methods
j = JSONFileReaderWriter2()
j.read("sample2.json")
j.write(data=['foo', {'bar': ('baz', None, 1.0, 2)}], filepath="sample2.json")

# Testing the TextFileReaderWriter
t = TextFileReaderWriter2()
t.write(filepath="sample.txt", data=["Line 1", "Line 2", "Line 3"])
t.read("sample.txt")
```

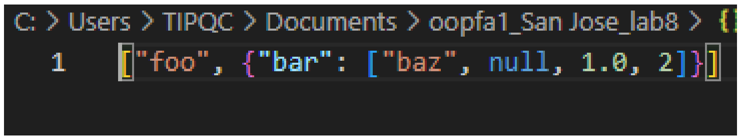


```python
from FileReaderWriter2 import FileReaderWriter2
from CSVFileReaderWriter2 import CSVFileReaderWriter2
from JSONFileReaderWriter2 import JSONFileReaderWriter2
from TextFileReaderWriter2 import TextFileReaderWriter2

# Testing default class
df = FileReaderWriter2()
df.read()
df.write()

# Testing the CSV methods
c = CSVFileReaderWriter2()
c.read("sample2.csv")
c.write(filepath="sample2.csv", data=["Hello", "World"])

# Testing the JSON methods
j = JSONFileReaderWriter2()
j.read("sample2.json")
j.write(data=['foo', {'bar': ('baz', None, 1.0, 2)}], filepath="sample2.json")

# Testing the TextFileReaderWriter
t = TextFileReaderWriter2()
t.write(filepath="sample2.txt", data=["Line 1", "Line 2", "Line 3"])
t.read("sample.txt")

```



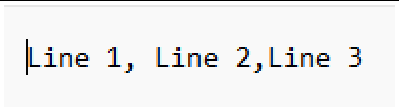
C: > Users > TIPQC > Documents > oop

```
Hello,World

```

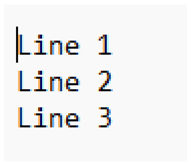


C: > Users > TIPQC > Documents > oopfa1_San Jose_lab8 > {}

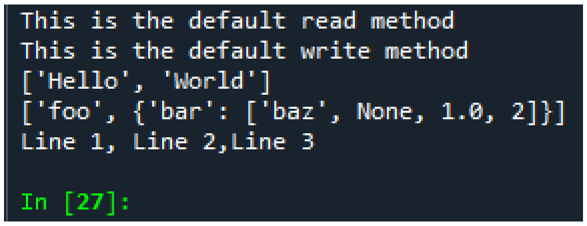
```
["foo", {"bar": ["baz", null, 1.0, 2]}]
```



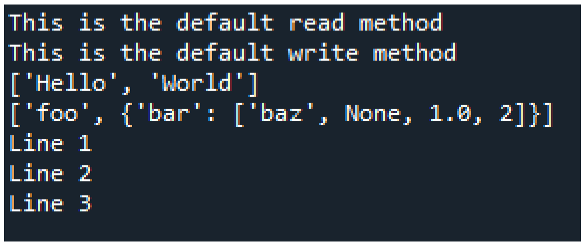
```
Line 1, Line 2,Line 3
```



```
Line 1
Line 2
Line 3
```



```
This is the default read method
This is the default write method
['Hello', 'World']
['foo', {'bar': ['baz', None, 1.0, 2]}]
Line 1, Line 2,Line 3

In [27]:
```



```
This is the default read method
This is the default write method
['Hello', 'World']
['foo', {'bar': ['baz', None, 1.0, 2]}]
Line 1
Line 2
Line 3
```

Questions

1. Why is Polymorphism important?

   Because polymorphism makes it easier to build programs that are flexible, scalable, and easy to maintain, so you can update or extend them with fewer headaches.

2. Explain the advantages and disadvantages of using applying Polymorphism in an Object-Oriented Program.
   - **Advantages**:
   - Code Reusability: You can reuse the same code for different types of objects, saving time and effort.
   - Flexibility: You can easily add new classes without breaking or changing existing code.
   - Easier Maintenance: Using a common interface makes it simpler to update and fix things in the program.
   - Dynamic Behavior: The program can pick the right method while it's running, making it more adaptable.
   - Better Readability: The code is cleaner and easier to understand for others.
   - **Disadvantages:**
   - Performance Issues: Polymorphism can slow down the program because it requires extra steps to determine which method to call.
   - Complex Implementation: Setting up polymorphism can be complicated and may confuse developers.
   - Debugging Problems: Errors can be harder to track down since multiple classes might override the same method.
   - Type Safety Risks: If not used carefully, you could encounter errors from using the wrong object type.
   - Learning Curve: It can be challenging for beginners to understand and use polymorphism effectively.

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

   - **Advantages:**

   - The program is simple to use since CSV is good for tables, and JSON handles structured data. Both formats are easy to read and understand, and can be used across different platforms. There’s also strong library support in most programming languages, making them easy to implement. Plus, they're lightweight, so they work well with small to medium-sized datasets.

   - **Disadvantages:**

   - However, performance can slow down with large datasets, especially compared to databases. CSV files have limited data types and don’t handle complex data like JSON. You also have to manually check for errors and ensure the data is valid. Handling large files can also be tricky when multiple users try to access them at the same time.

-

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

   When using polymorphism, start by setting clear goals like improving code reuse or making maintenance easier. Carefully plan your class structure, ensuring related classes share a common interface or superclass. Be mindful of type safety, performance, and keep your code readable, while testing thoroughly to avoid unexpected issues.

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

Polymorphism is really helpful in many everyday programs. It allows different objects, like UI elements, game characters, or user types, to be handled in the same way, even if they have unique features. This makes coding more flexible, easier to manage, and more organized, which is crucial for modern software development.

7. Conclusion:

In our discussion, we examined polymorphism in object-oriented programming by creating a base class called **FileReaderWriter** and subclasses for handling CSV, JSON, and text files. This illustrated how different classes can have unique methods while still sharing a common interface, making the code cleaner and easier to manage. Overall, we highlighted how polymorphism enhances organization and efficiency in programming, especially when working with various file types.

8. Assessment Rubric: