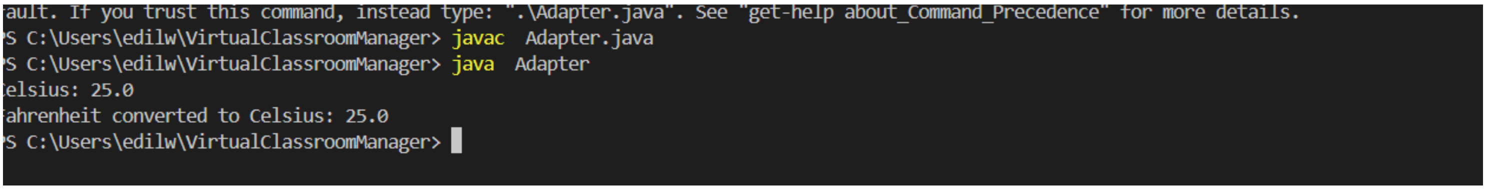
Project Overview

ROVER OUTPUT SNAPSHOTS

ADAPTER PATTERN

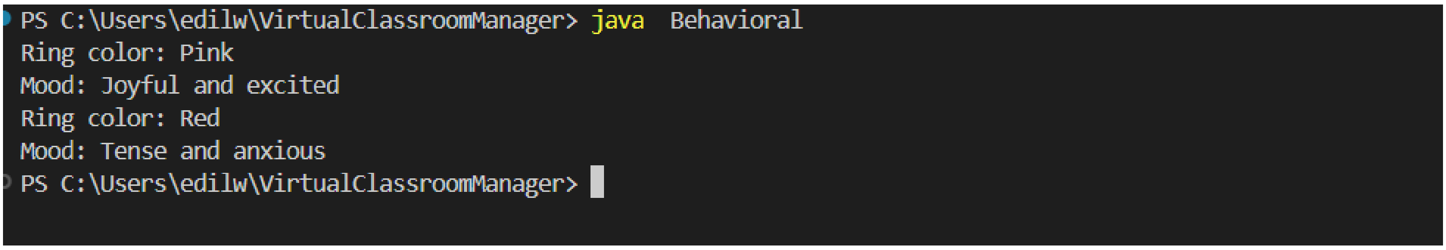
The provided code effectively demonstrates the Adapter pattern by bridging the gap between two incompatible interfaces. The FahrenheitToCelsiusAdapter adapts the FahrenheitThermometer to the CelsiusTemperature



```
ault. If you trust this command, instead type: ".\Adapter.java". See "get-help about_Command_Precedence" for more details.
S C:\Users\edilw\VirtualClassroomManager> javac  Adapter.java
S C:\Users\edilw\VirtualClassroomManager> java  Adapter
elsius: 25.0
ahrenheit converted to Celsius: 25.0
S C:\Users\edilw\VirtualClassroomManager> 
```

BEHAVIORAL PATTERN

By encapsulating the different mood behaviors within separate objects and allowing the MoodRing to delegate behavior based on its current state, the code effectively implements the State pattern, a behavioral design pattern.



```
PS C:\Users\edilw\VirtualClassroomManager> java  Behavioral
Ring color: Pink
Mood: Joyful and excited
Ring color: Red
Mood: Tense and anxious
PS C:\Users\edilw\VirtualClassroomManager> 
```

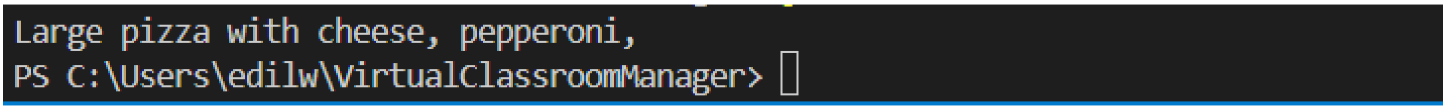
BUILDER PATTERN

Complex object creation: The Pizza class represents a complex object with multiple attributes.

Builder class: The PizzaBuilder class is responsible for constructing the Pizza object step by step.

Fluent interface: The builder methods return the builder itself, allowing for chaining of method calls.

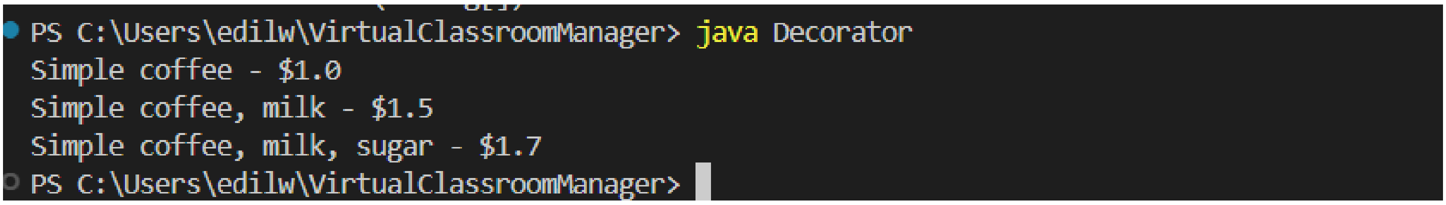
Separation of concerns: The builder handles the construction process, while the Pizza class focuses on representing the product.



```
Large pizza with cheese, pepperoni,
PS C:\Users\edilw\VirtualClassroomManager> 
```

DECORATOR PATTERN

In essence, the provided code effectively implements the Decorator pattern to create different coffee variations by adding ingredients dynamically.



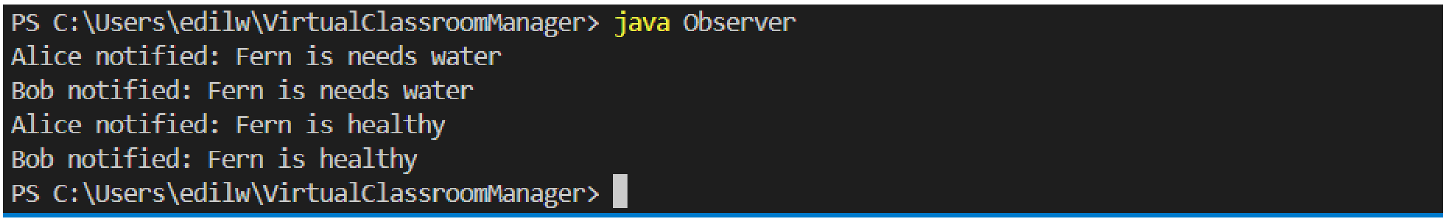
```
PS C:\Users\edilw\VirtualClassroomManager> java Decorator
Simple coffee - $1.0
Simple coffee, milk - $1.5
Simple coffee, milk, sugar - $1.7
PS C:\Users\edilw\VirtualClassroomManager> 
```

FACTORY PTTERN

By using the Factory Method pattern, you've achieved a flexible and maintainable design for creating different types of emojis.

OBSERVER PATTERN

Code effectively uses the Observer pattern to create a system where plants can notify their respective owners about their condition.



```
PS C:\Users\edilw\VirtualClassroomManager> java Observer
Alice notified: Fern is needs water
Bob notified: Fern is needs water
Alice notified: Fern is healthy
Bob notified: Fern is healthy
PS C:\Users\edilw\VirtualClassroomManager> 
```