1. Command Pattern Implementation:

Purpose: The Command Pattern separates command execution from the entity invoking it. It encapsulates actions as objects, enabling parameterization and queuing of commands.

In the Code: The `Command` base class defines a blueprint for various actions (`StartChecksCommand`, `LaunchCommand`, `FastForwardCommand`) to be executed on the `Rocket`. Each command encapsulates a specific action for the rocket.

2. Observer Pattern Implementation:

Purpose: The Observer Pattern establishes a relationship between objects, notifying dependent objects when a subject's state changes.

In the Code: The Observable class maintains a list of observers (basic implementation), allowing them to track changes in the rocket's state. The Rocket class inherits from Observable, theoretically enabling observers to monitor the rocket's state changes.

3. Detailed Overview of the Rocket Class:

Attributes: The Rocket class holds attributes such as stage, fuel, altitude, and speed, representing different aspects of the rocket's state during its launch.

Methods: Several methods in the Rocket class simulate various stages of the rocket's launch process:

start_checks: Sets up parameters required for the rocket's launch.

launch: Initiates the rocket's launch sequence.

update_status_continuously: Continuously updates the rocket's status until certain conditions, like reaching an altitude or running out of fuel, are met.

fast_forward: Simulates the rocket's progress through time, modifying altitude, speed, and fuel based on a given duration.

4. User Interface and Parsing:

user_interface Function: Manages user interaction by collecting commands to interact with the rocket.

parse_input Function: Translates user commands into specific command objects that the rocket understands and executes.

5. Key Design Decisions and Patterns:

Command Modularity: Commands encapsulate individual actions, ensuring flexibility without altering the main Rocket class.

Observer Pattern for Extensibility: The code allows potential addition of observers to monitor the rocket's state changes without affecting its behavior directly.

Basic Error Handling: It provides simple guidance for incorrect inputs to improve user interaction.

6. Conclusion:

The code sets up a structured rocket launch simulation using command and observer patterns. It provides a method for interacting with the rocket and simulates its behavior throughout a launch process. Users can input parameters such as altitude threshold (target), altitude increase per second or iteration (for the rocket), speed per iteration, and fuel decrease per iteration, granting maximum control over simulation aspects like distance, fuel, and speed.