**Exercise 3: Implementing the Builder Pattern**

**Scenario:**

You are developing a system to create complex objects such as a Computer with multiple optional parts. Use the Builder Pattern to manage the construction process.

**Steps:**

1. **Create a New Java Project:**
   * Create a new Java project named **BuilderPatternExample**.
2. **Define a Product Class:**
   * Create a class **Computer** with attributes like **CPU**, **RAM**, **Storage**, etc.
3. **Implement the Builder Class:**
   * Create a static nested Builder class inside Computer with methods to set each attribute.
   * Provide a **build()** method in the Builder class that returns an instance of Computer.
4. **Implement the Builder Pattern:**
   * Ensure that the **Computer** class has a private constructor that takes the **Builder** as a parameter.
5. **Test the Builder Implementation:**

Create a test class to demonstrate the creation of different configurations of Computer using the Builder pattern.

**Script:**

**Computer.java:**

public class Computer {

    private String CPU;

    private String RAM;

    private String storage;

    private String graphicsCard;

    private String keyboard;

    private String monitor;

    private Computer(Builder builder) {

        this.CPU = builder.CPU;

        this.RAM = builder.RAM;

        this.storage = builder.storage;

        this.graphicsCard = builder.graphicsCard;

        this.keyboard = builder.keyboard;

        this.monitor = builder.monitor;

    }

    public void displayConfig() {

        System.out.println("Computer Config:");

        System.out.println("CPU: " + CPU);

        System.out.println("RAM: " + RAM);

        System.out.println("Storage: " + (storage != null ? storage : "None"));

        System.out.println("Graphics Card: " + (graphicsCard != null ? graphicsCard : "None"));

        System.out.println("Keyboard: " + (keyboard != null ? keyboard : "None"));

        System.out.println("Monitor: " + (monitor != null ? monitor : "None"));

    }

    public static class Builder {

        private String CPU;

        private String RAM;

        private String storage;

        private String graphicsCard;

        private String keyboard;

        private String monitor;

        public Builder(String CPU, String RAM) {

            this.CPU = CPU;

            this.RAM = RAM;

        }

        public Builder setStorage(String storage) {

            this.storage = storage;

            return this;

        }

        public Builder setGraphicsCard(String graphicsCard) {

            this.graphicsCard = graphicsCard;

            return this;

        }

        public Builder setKeyboard(String keyboard) {

            this.keyboard = keyboard;

            return this;

        }

        public Builder setMonitor(String monitor) {

            this.monitor = monitor;

            return this;

        }

        public Computer build() {

            return new Computer(this);

        }

    }

}

**Main.java:**

public class Main {

    public static void main(String[] args) {

        Computer basicComputer = new Computer.Builder("Intel i5", "8GB").build();

        System.out.println("\n Basic Computer:");

        basicComputer.displayConfig();

        Computer gamingComputer = new Computer.Builder("AMD Ryzen 9", "32GB")

            .setGraphicsCard("NVIDIA RTX 4080")

            .setStorage("1TB SSD")

            .setKeyboard("Mechanical RGB Keyboard")

            .setMonitor("27-inch 4K Monitor")

            .build();

        System.out.println("\n Gaming Computer:");

        gamingComputer.displayConfig();

        Computer wfhComputer = new Computer.Builder("Intel i7", "16GB")

            .setStorage("512GB SSD")

            .setKeyboard("Wireless Keyboard")

            .build();

        System.out.println("\n Work From Home Computer:");

        wfhComputer.displayConfig();

    }

}