**Module 1 - Design Patterns and Principles**

**Exercise 3: Implementing the Builder Pattern**

**Aim:**

To develop a system to create complex objects such as a Computer with multiple optional parts using the Builder Pattern to manage the construction process.

**Code:**

class Computer {

    private String cpu;

    private String ram;

    private String storage;

    private Computer(Builder builder) {

        this.cpu = builder.cpu;

        this.ram = builder.ram;

        this.storage = builder.storage;

    }

    public String toString() {

        return "Computer Configuration: CPU=" + cpu + ", RAM=" + ram + ", Storage=" + storage;

    }

    public static class Builder {

        private String cpu;

        private String ram;

        private String storage;

        public Builder setCpu(String cpu) {

            this.cpu = cpu;

            return this;

        }

        public Builder setRam(String ram) {

            this.ram = ram;

            return this;

        }

        public Builder setStorage(String storage) {

            this.storage = storage;

            return this;

        }

        public Computer build() {

            return new Computer(this);

        }

    }

}

public class BuilderPatternExample {

    public static void main(String[] args) {

        Computer c1 = new Computer.Builder()

            .setCpu("i7")

            .setRam("16GB")

            .setStorage("512GB")

            .build();

        System.out.println(c1);

    }

}

**Output:**

