**Exercise 3: Implementing the Builder Pattern**

CODE:

public class BuilderPatternExample {

static class Computer {

private String CPU;

private String RAM;

private String storage;

private String graphicsCard;

private Computer(Builder builder) {

this.CPU = builder.CPU;

this.RAM = builder.RAM;

this.storage = builder.storage;

this.graphicsCard = builder.graphicsCard;

}

@Override

public String toString() {

return "Computer [CPU=" + CPU + ", RAM=" + RAM +

", Storage=" + storage + ", GraphicsCard=" + graphicsCard + "]";

}

public static class Builder {

private String CPU;

private String RAM;

private String storage;

private String graphicsCard;

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 Computer build() {

return new Computer(this);

}

}

}

public static void main(String[] args) {

Computer basicComputer = new Computer.Builder("Intel i3", "8GB").build();

Computer gamingComputer = new Computer.Builder("Intel i9", "32GB")

.setStorage("1TB SSD")

.setGraphicsCard("NVIDIA RTX 4080")

.build();

Computer officeComputer = new Computer.Builder("AMD Ryzen 5", "16GB")

.setStorage("512GB SSD")

.build();

System.*out*.println("Basic Computer: " + basicComputer);

System.*out*.println("Gaming Computer: " + gamingComputer);

System.*out*.println("Office Computer: " + officeComputer);

}

}

OUTPUT:

