**Builder Design Pattern**

**Assignment - 1**

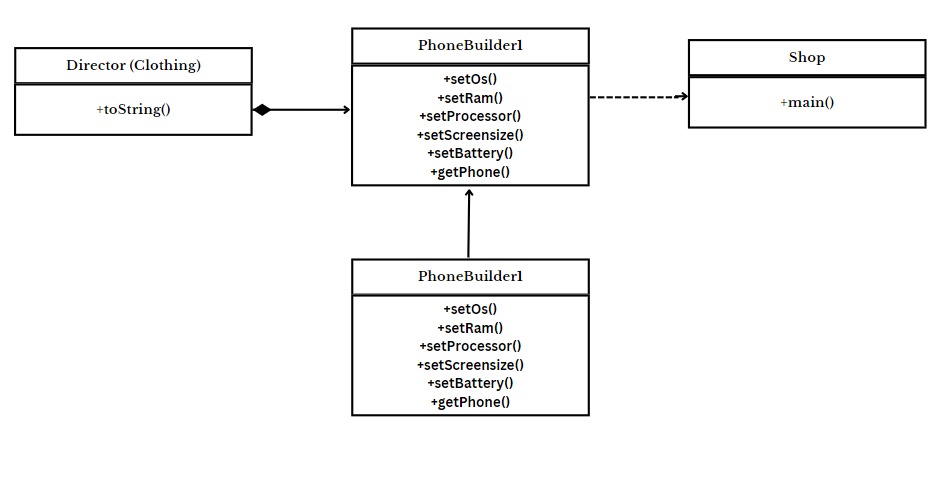
Name: Heet Dobariya Roll No. - 22BCP177 Group - G5

* **Builder Design :**

The Builder Design Pattern is a creational design pattern that provides a way to construct a complex object step by step. It separates the construction of a complex object from its representation, allowing the same construction process to create different representations.

The main idea behind the Builder pattern is to have a dedicated "Builder" class responsible for constructing an object with a complex structure. This builder class typically has a set of methods for configuring different parts of the object. There is also a "Director" class that orchestrates the construction process by using the builder to create the final object.

* **Program :** Create a Builder Design Pattern for Phone example.
* **UML Diagram :**



* **Code :**

package PhoneBuilder;

public class Phone

{

private String os;

private int ram;

private String processor;

private double screensize;

private int battery;

public Phone(String os, int ram, String processor, double screensize, int battery)

{

this.os = os;

this.ram = ram;

this.processor = processor;

this.screensize = screensize;

this.battery = battery;

}

@Override

public String toString()

{

return "Phone [OS ="+os+", RAM ="+ram+", Processor =" +processor+", Screensize ="+screensize+", Battery ="+battery+"]";

}

}

package PhoneBuilder;

public class PhoneBuilder

{

private String os;

private int ram;

private String processor;

private double screensize;

private int battery;

public PhoneBuilder setOs(String os1)

{

this.os = os1;

return this;

}

public PhoneBuilder setRam(int ram1)

{

this.ram = ram1;

return this;

}

public PhoneBuilder setProcessor(String processor1)

{

this.processor = processor1;

return this;

}

public PhoneBuilder setScreensize(double screensize1)

{

this.screensize = screensize1;

return this;

}

public PhoneBuilder setBattery(int battery1)

{

this.battery = battery1;

return this;

}

public Phone getPhone()

{

return new Phone(os, ram, processor, screensize, battery);

}

}

package PhoneBuilder;

public class PhoneBuilder1

{

private String os;

private int ram;

private String processor;

private double screensize;

private int battery;

public void setOs(String os1)

{

this.os = os1;

}

public void setRam(int ram1)

{

this.ram = ram1;

}

public void setProcessor(String processor1)

{

this.processor = processor1;

}

public void setScreensize(double screensize1)

{

this.screensize = screensize1;

}

public void setBattery(int battery1)

{

this.battery = battery1;

}

public Phone getPhone()

{

return new Phone(os, ram, processor, screensize, battery);

}

}

package PhoneBuilder;

public class Shop

{

public static void main(String[] args)

{

Phone p = new PhoneBuilder().setOs("Android").setRam(2).setBattery(3100).getPhone();

System.out.println(p);

// You can set the no. of parameters by yourself sequence does not matter, set them in any order

// If you don't want the above cascaded calls for Phone Object creation

// then create PhoneBuilder1 class object and execute the setters call separately

PhoneBuilder1 p1 = new PhoneBuilder1();

p1.setOs("Android");

p1.setRam(16);

p1.setBattery(5000);

System.out.println(p1.getPhone());

}

}

* **Output :**

