**Factory Pattern Assignment - 1**

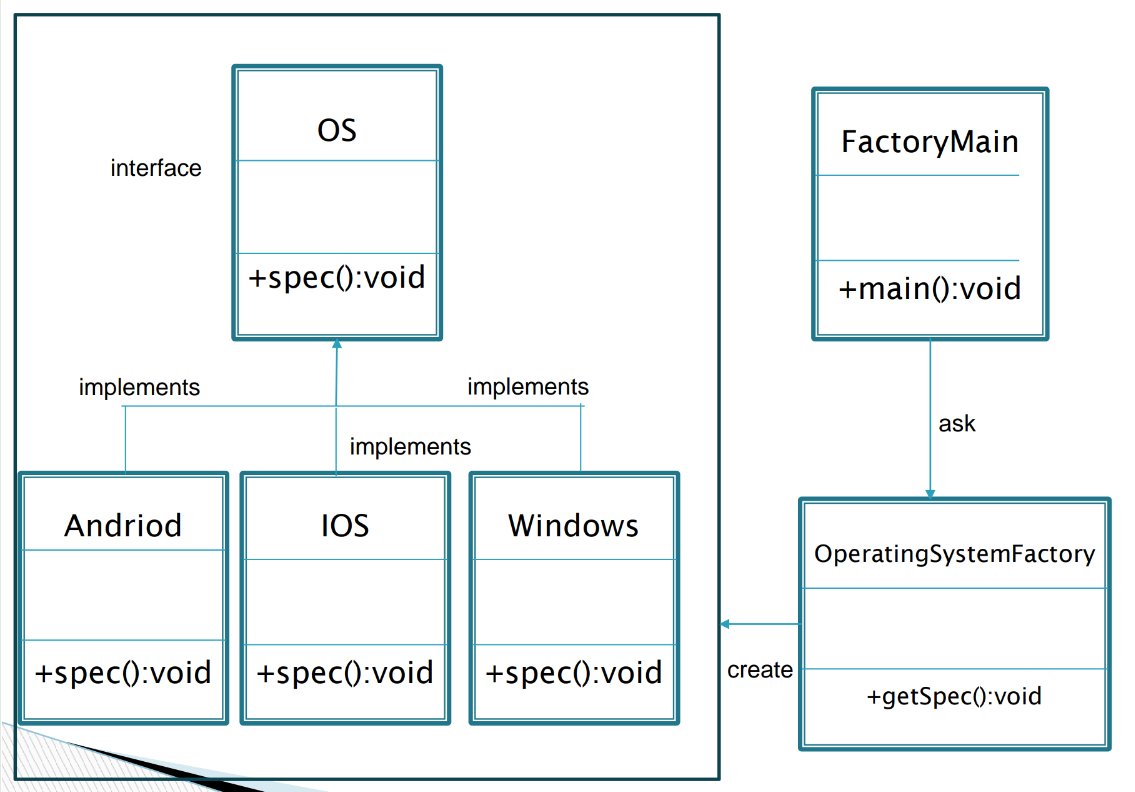
Name: Heet Dobariya Roll No. - 22BCP177 Group - G5

* **Factory Design :**

A creational design pattern called the Factory Design Pattern gives subclasses the ability to modify the kind of objects that are created while still providing an interface for doing so in a super class. It entails specifying an interface for object creation, allowing subclasses to choose the type of object to be created, and then constructing the instance in a method.

By abstracting the object creation process, this design facilitates loose coupling by facilitating changes or extensions to the creation logic without requiring changes to the client code.

* **Program :** Create a Factory Design Pattern for Mobile Factory example.
* **UML Diagram :**



* **Code :**

public interface OS

{

    public void spec();

}

public class Android implements OS

{

    @ Override

    public void spec()

    {

        System.out.println("Most Powerful OS");

    }

}

public class IOS implements OS

{

    @ Override

    public void spec()

    {

        System.out.println("Most Secured OS");

    }

}

public class Windows implements OS

{

    @ Override

    public void spec()

    {

        System.out.println("I am about to die!");

    }

}

public class OperatingSystemFactory

{

    public OS getInstance(String str)

    {

        if (str.equals("Open"))

            return new Android();

        else if (str.equals("Closed"))

            return new IOS();

        else

            return new Windows();

    }

}

public class FactoryMain

{

    public static void main(String[] args)

    {

        OperatingSystemFactory osf = new OperatingSystemFactory();

        OS obj = osf.getInstance("Closed");

        obj.spec();

    }

}

* **Output :**

Screenshot (222)