Implement a simple Shape factory with Shape as abstract product.

Step1: Create a new project with name ShapeSimpleFactoryApp

Step 2: Create an interface for abstract product

|  |
| --- |
| public interface IShape  {  void Draw();  } |

Step3:Create multiple products from the interface

|  |
| --- |
| public class Rectangle : IShape  {  public void Draw()  {  Console.WriteLine("Inside Rectangle::draw() method.");  }  }  public class Square : IShape  {  public void Draw()  {  Console.WriteLine("Inside Square::draw() method.");  }  } |

Step 4:Create a simple factory class as shown below

|  |
| --- |
| public class ShapeFactory  {    public IShape Getshape(String shapeType)  {  if (shapeType == null)  {  return null;  }  if (shapeType.ToUpper().Equals("CIRCLE"))  {  return new Circle();  }  else if (shapeType.Equals("RECTANGLE"))  {  return new Rectangle();  }  else if (shapeType.Equals("SQUARE"))  {  return new Square();  }  return null;  }  } |
|  |

Step5:Complete the Main method

|  |
| --- |
| namespace SimpleFactoryLab  {  internal class Program  {  static void Main(string[] args)  {  ShapeFactory shapeFactory = new ShapeFactory();  //get an object of Circle and call its draw method.  IShape shape1 = shapeFactory.Getshape("CIRCLE");  //call draw method of Circle  shape1.Draw();  //get an object of Rectangle and call its draw method.  IShape shape2 = shapeFactory.Getshape("RECTANGLE");  //call draw method of Rectangle  shape2.Draw();  //get an object of Square and call its draw method.  IShape shape3 = shapeFactory.Getshape("SQUARE");  //call draw method of square  shape3.Draw();  }  }  } |

//