Exercise 7: Financial Forecasting:

using System;

namespace FinancialForecastFactory

{

    // Product Interface

    interface IForecast

    {

        void GenerateForecast();

    }

    // Concrete Product: Sales Forecast

    class SalesForecast : IForecast

    {

        public void GenerateForecast()

        {

            Console.WriteLine("Generating sales forecast: Expected revenue increase by 10%.");

        }

    }

    // Concrete Product: Expense Forecast

    class ExpenseForecast : IForecast

    {

        public void GenerateForecast()

        {

            Console.WriteLine("Generating expense forecast: Operational costs may rise by 5%.");

        }

    }

    // Concrete Product: Profit Forecast

    class ProfitForecast : IForecast

    {

        public void GenerateForecast()

        {

            Console.WriteLine("Generating profit forecast: Net profit expected to grow by 8%.");

        }

    }

    // Creator Abstract Class

    abstract class ForecastEngine

    {

        public abstract IForecast CreateForecast();

        public void RunForecast()

        {

            var forecast = CreateForecast();

            forecast.GenerateForecast();

        }

    }

    // Concrete Creators

    class SalesForecastEngine : ForecastEngine

    {

        public override IForecast CreateForecast()

        {

            return new SalesForecast();

        }

    }

    class ExpenseForecastEngine : ForecastEngine

    {

        public override IForecast CreateForecast()

        {

            return new ExpenseForecast();

        }

    }

    class ProfitForecastEngine : ForecastEngine

    {

        public override IForecast CreateForecast()

        {

            return new ProfitForecast();

        }

    }

    // Main Program

    class Program

    {

        static void Main(string[] args)

        {

            Console.WriteLine("Forecast type (sales/expense/profit):");

            string type = Console.ReadLine() ?? string.Empty;

            if (string.IsNullOrWhiteSpace(type))

            {

                Console.WriteLine("Forecast type cannot be empty.");

                return;

            }

            ForecastEngine engine;

            switch (type.ToLower())

            {

                case "sales":

                    engine = new SalesForecastEngine();

                    break;

                case "expense":

                    engine = new ExpenseForecastEngine();

                    break;

                case "profit":

                    engine = new ProfitForecastEngine();

                    break;

                default:

                    Console.WriteLine("Invalid forecast type.");

                    return;

            }

            engine.RunForecast();

        }

    }

}

OUTPUT:

