C# OOP Case Studies (Beginner to Intermediate)

This document contains 10 complete case studies demonstrating core Object-Oriented Programming concepts in C#. Each example is self-contained and includes full working code.

CaseStudy1_Student

```
using System;

class Student
{
    public string Name;
    public int Age;
    public string Grade;

    public void DisplayInfo()
    {
        Console.WriteLine($"Name: {Name}, Age: {Age}, Grade: {Grade}");
    }
}

class Program
{
    static void Main()
    {
        Student student1 = new Student();
        student1.Name = "John";
        student1.Age = 20;
        student1.Grade = "A";
        student1.DisplayInfo();
    }
}
```

CaseStudy2_BankAccount

```
using System;

class BankAccount
{
    private double balance;

    public void Deposit(double amount)
    {
        if (amount > 0)
            balance += amount;
    }

    public void Withdraw(double amount)
    {
        if (amount > 0 && amount <= balance)
            balance -= amount;
        else
            Console.WriteLine("Invalid Withdrawal");
    }

    public double GetBalance()
    {
        return balance;
    }
}

class Program
{
    static void Main()
    {
        BankAccount account = new BankAccount();
        account.Deposit(500);
        account.Withdraw(200);
        Console.WriteLine("Balance: $" + account.GetBalance());
    }
}
```

CaseStudy3_Employee

```
using System;

class Employee
{
    public string Name;
    public double Salary;

    public void Display()
    {
        Console.WriteLine($"Name: {Name}, Salary: {Salary}");
    }
}

class Manager : Employee
{
    public string Department;

    public void DisplayManager()
    {
        Display();
        Console.WriteLine($"Department: {Department}");
    }
}

class Program
{
    static void Main()
    {
        Manager manager = new Manager();
        manager.Name = "Alice";
        manager.Salary = 90000;
        manager.Department = "IT";
        manager.DisplayManager();
    }
}
```

CaseStudy4_ShapePolymorphism

```
using System;

class Shape
{
    public virtual void Draw()
    {
        Console.WriteLine("Drawing a shape...");
    }
}

class Circle : Shape
{
    public override void Draw()
    {
        Console.WriteLine("Drawing a Circle");
    }
}

class Rectangle : Shape
{
    public override void Draw()
    {
        Console.WriteLine("Drawing a Rectangle");
    }
}

class Program
{
    static void Main()
    {
        Shape shape;
        shape = new Circle();
        shape.Draw();

        shape = new Rectangle();
        shape.Draw();
    }
}
```

CaseStudy5_PaymentSystem

```
using System;

interface IPayment
{
    void MakePayment(double amount);
}

class CreditCard : IPayment
{
    public void MakePayment(double amount)
    {
        Console.WriteLine($"Paid ${amount} using Credit Card.");
    }
}

class PayPal : IPayment
{
    public void MakePayment(double amount)
    {
        Console.WriteLine($"Paid ${amount} using PayPal.");
    }
}

class Program
{
    static void Main()
    {
        IPayment payment;

        payment = new CreditCard();
        payment.MakePayment(150.00);

        payment = new PayPal();
        payment.MakePayment(200.00);
    }
}
```

CaseStudy6_ProductInventory

```
using System;

class Product
{
    public string Name;
    public double Price;

    public Product()
    {
        Name = "Unknown";
        Price = 0.0;
    }

    public Product(string name, double price)
    {
        Name = name;
        Price = price;
    }

    public void Display()
    {
        Console.WriteLine($"Product: {Name}, Price: ${Price}");
    }
}

class Program
{
    static void Main()
    {
        Product product1 = new Product();
        Product product2 = new Product("Laptop", 1200.99);
        product1.Display();
        product2.Display();
    }
}
```

CaseStudy7_LibraryBook

```
using System;

class Book
{
    public string Title;
    public string Author;

    public Book(string title, string author)
    {
        Title = title;
        Author = author;
    }

    public void Display()
    {
        Console.WriteLine($"Book: {Title}, Author: {Author}");
    }
}

class Library
{
    public string Name;
    public Book Book;

    public Library(string name, Book book)
    {
        Name = name;
        Book = book;
    }

    public void DisplayLibrary()
    {
        Console.WriteLine($"Library: {Name}");
        Book.Display();
    }
}

class Program
{
    static void Main()
    {
        Book book = new Book("1984", "George Orwell");
        Library library = new Library("City Library", book);
        library.DisplayLibrary();
    }
}
```

CaseStudy8_CarFactory

```
using System;

class Car
{
    public string Model;
    public static int CarCount = 0;

    public Car(string model)
    {
        Model = model;
        CarCount++;
    }

    public static void ShowCarCount()
    {
        Console.WriteLine($"Total Cars Produced: {CarCount}");
    }
}

class Program
{
    static void Main()
    {
        Car car1 = new Car("Tesla Model S");
        Car car2 = new Car("Ford Mustang");
        Car car3 = new Car("BMW M3");
        Car.ShowCarCount();
    }
}
```

CaseStudy9_AnimalAbstract

```
using System;

abstract class Animal
{
    public abstract void MakeSound();
}

class Dog : Animal
{
    public override void MakeSound()
    {
        Console.WriteLine("Dog says: Woof!");
    }
}

class Cat : Animal
{
    public override void MakeSound()
    {
        Console.WriteLine("Cat says: Meow!");
    }
}

class Program
{
    static void Main()
    {
        Animal animal;

        animal = new Dog();
        animal.MakeSound();

        animal = new Cat();
        animal.MakeSound();
    }
}
```

CaseStudy10_VehicleInterface

```
using System;

interface IVehicle
{
    void Drive();
}

class Car : IVehicle
{
    public void Drive()
    {
        Console.WriteLine("Driving a car...");
    }
}

class Bike : IVehicle
{
    public void Drive()
    {
        Console.WriteLine("Riding a bike...");
    }
}

class Program
{
    static void Main()
    {
        IVehicle vehicle;

        vehicle = new Car();
        vehicle.Drive();

        vehicle = new Bike();
        vehicle.Drive();
    }
}
```