**Best Programming Practices**

* Use meaningful class names (e.g., Student, Employee) and method names (e.g., DisplayDetails).
* Encapsulate data using private fields and provide getter and setter methods.
* Follow proper naming conventions (camelCase for attributes and methods).
* Always provide constructors to initialize class attributes.
* Use comments for clarity and better readability.

**Sample Program 1: Food Delivery App**

**Real-World Analogy**

Imagine a food delivery app like Swiggy or Uber Eats. The app deals with restaurants, and each restaurant has specific details like its name, location, and the food items it serves.

**Step 1: Define the Class**

The Restaurant class represents the blueprint for creating restaurant objects.

// Class Definition

using System;

public class Restaurant

{

// Fields (Attributes)

private string name;

private string location;

private string[] foodItems;

// Constructor

public Restaurant(string name, string location, string[] foodItems)

{

this.name = name;

this.location = location;

this.foodItems = foodItems;

}

// Method to display restaurant details

public void DisplayDetails()

{

Console.WriteLine("Restaurant Name: " + name);

Console.WriteLine("Location: " + location);

Console.WriteLine("Food Items:");

foreach (string item in foodItems)

{

Console.WriteLine("- " + item);

}

}

// Method to check if a food item is available

public bool IsFoodAvailable(string food)

{

foreach (string item in foodItems)

{

if (item.Equals(food, StringComparison.OrdinalIgnoreCase))

{

return true;

}

}

return false;

}

}



**Step 2: Create Objects from the Class**

Use the class to create specific restaurant objects.

// Main Class to Test

public class Program

{

public static void Main(string[] args)

{

// Define food items for restaurants

string[] foodItems1 = { "Pizza", "Pasta", "Burger" };

string[] foodItems2 = { "Sushi", "Ramen", "Tempura" };

// Create Restaurant objects

Restaurant restaurant1 = new Restaurant("Italian Delight", "Downtown", foodItems1);

Restaurant restaurant2 = new Restaurant("Tokyo Treats", "Uptown", foodItems2);

// Display details of each restaurant

Console.WriteLine("=== Restaurant 1 ===");

restaurant1.DisplayDetails();

Console.WriteLine("\n=== Restaurant 2 ===");

restaurant2.DisplayDetails();

// Check food availability

Console.WriteLine("\nChecking Food Availability:");

Console.WriteLine("Is Pasta available in Italian Delight? " + restaurant1.IsFoodAvailable("Pasta"));

Console.WriteLine("Is Sushi available in Italian Delight? " + restaurant1.IsFoodAvailable("Sushi"));

}

}



**Step 3: Output**

=== Restaurant 1 ===

Restaurant Name: Italian Delight

Location: Downtown

Food Items:

- Pizza

- Pasta

- Burger

=== Restaurant 2 ===

Restaurant Name: Tokyo Treats

Location: Uptown

Food Items:

- Sushi

- Ramen

- Tempura

Checking Food Availability:

Is Pasta available in Italian Delight? True

Is Sushi available in Italian Delight? False

**In-depth Explanation of Key Aspects**

1. **Fields (Attributes)**
   * Fields store the data for the class.
   * Example: name, location, and foodItems represent the state of a restaurant.
2. **Constructor**
   * A constructor initializes the fields when an object is created.
   * Example: The Restaurant constructor sets name, location, and foodItems.
3. **Methods**
   * Methods define the behavior of the objects.
   * Example:
     + DisplayDetails(): Displays the details of a restaurant.
     + IsFoodAvailable(string food): Checks if a specific food item is available.
4. **Encapsulation**
   * The fields are marked as private and accessed using methods to ensure controlled data access and modification.
5. **Object Creation**
   * Objects are created using the new keyword.
   * Example:
   * Restaurant restaurant1 = new Restaurant("Italian Delight", "Downtown", foodItems1);
6. **Memory Allocation**
   * Each object has its own memory space for attributes but shares methods.

Level 1 Practice Programs

**1. Program to Display Employee Details**

**Problem Statement:** Write a program to create an Employee class with attributes name, id, and salary. Add a method to display the details.

**Step 1: Define the Class**

| **using System; using System.Collections.Generic; using System.Linq; using System.Text; using System.Threading.Tasks;  namespace EmployeeApp {  public class Employee  {  // Fields (Attributes)**  **private string name;  private int id;  private double salary;**  **// Constructor  public Employee(string name, int id, double salary)  {  this.name = name;  this.id = id;  this.salary = salary;  }**  **// Method to display employee details  public void DisplayDetails()  {  Console.WriteLine("Employee Name: " + name);  Console.WriteLine("Employee ID: " + id);  Console.WriteLine("Salary: " + salary);  }  } }** |
| --- |

**Step 2: Create Objects from the Class**

| **using System;  namespace EmployeeApp {  public class Program  {  public static void Main(string[] args)  {  // create Employee Class object   Employee emp = new Employee("Rohit Dixit", 174, 50000);**  **// display result  emp.DisplayDetails();  }  } }** |
| --- |

**2. Program to Compute Area of a Circle**

**Problem Statement:** Write a program to create a Circle class with an attribute radius. Add methods to calculate and display the area and circumference of the circle.

**Step 1: Define the Class**

| **using System; using System.Collections.Generic; using System.Linq; using System.Text; using System.Threading.Tasks;  namespace AreaOfCircle {  public class Circle  {  // Fields (Attributes)  private double radius;    // constructor  public Circle(double radius)  {  this.radius = radius;  }   // method to calculate area  public double CalculateArea()  {  return Math.PI \* radius \* radius;  }   // method to calculate circumference  public double CalculateCircumference()  {  return 2 \* Math.PI \* radius;  }   // method to display result  public void DisplayDetails()  {  Console.WriteLine("Radius: " + radius);  Console.WriteLine("Area: " + CalculateArea());  Console.WriteLine("Circumference: " + CalculateCircumference());  }  } }** |
| --- |

**Step 2: Create Objects from the Class**

| **using System;  namespace AreaOfCircle {**  **class Program  {  public static void Main(string[] args)  {  // create Circle class object  Circle circle = new Circle(5);   // display result  circle.DisplayDetails();   }  } }** |
| --- |

**3. Program to Handle Book Details**

**Problem Statement:** Write a program to create a Book class with attributes title, author, and price. Add a method to display the book details.

**Step 1: Define the Class**

| **using System; using System.Collections.Generic; using System.Linq; using System.Text; using System.Threading.Tasks;  namespace BookDetails {  public class Book  {  // Fields (Attributes)  private string title;  private string author;  private double price;   // constructor  public Book(string title, string author, double price)  {  this.title = title;  this.author = author;  this.price = price;  }   // method to display result  public void DisplayDetails()  {  Console.WriteLine("Book Title: " + title);  Console.WriteLine("Author: " + author);  Console.WriteLine("Price: " + price);  }  } }** |
| --- |

**Step 2: Create Objects from the Class**

| **using System;  namespace BookDetails {  class Program   {  public static void Main(string[] args)  {  // create Book class object  Book book = new Book("C# Programming", "Rohit Dixit", 100);   // display result  book.DisplayDetails();  }  } }** |
| --- |