Set C Solution

1. **Create a class Product with attributes ProductID, ProductName, and Price. Write a constructor to initialize these attributes and a method DisplayProductDetails() to print these details.**

```
using System;

namespace ProductApp
{
   public class Product
   {
      // Attributes
      public int ProductID { get; set; }
      public string ProductName { get; set; }
      public decimal Price { get; set; }

      // Constructor to initialize attributes
      public Product(int productID, string productName, decimal price)
      {
         ProductID = productID;
         ProductName = productName;
         Price = price;
      }

      // Method to display product details
      public void DisplayProductDetails()
      {
         Console.WriteLine("Product Details:");
         Console.WriteLine($"Product ID: {ProductID}");
         Console.WriteLine($"Product Name: {ProductName}");
         Console.WriteLine($"Price: {Price:C}");
      }

      // Entry point
      public static void Main(string[] args)
      {
         // Create a Product object
         Product product = new Product(101, "Laptop", 75000.50m);

         // Display the product details
         product.DisplayProductDetails();
      }
   }
}
```

2. Create a class BankTransaction with methods to process transactions in multiple ways:

- **Deposit(amount: double) - Deposits a specified amount.**
- **Deposit(amount: double, chequeNo: int) - Deposits using a cheque.**
- **Deposit(amount: double, cardNo: string) - Deposits using a credit/debit card.**

```
using System;

namespace BankTransactionApp
{
    public class BankTransaction
    {
        public double Balance = 0;

        // Method to deposit a specified amount
        public void Deposit(double amount)
        {
            if (amount <= 0)
            {
                Console.WriteLine("Invalid amount. Deposit failed.");
                return;
            }
            Balance += amount;
            Console.WriteLine($"Deposited {amount:C} in cash. New balance: {Balance:C}");
        }

        // Method to deposit a specified amount using a cheque
        public void Deposit(double amount, int chequeNo)
        {
            if (amount <= 0)
            {
                Console.WriteLine("Invalid amount. Deposit failed.");
                return;
            }
            if (chequeNo <= 0)
            {
                Console.WriteLine("Invalid cheque number. Deposit failed.");
                return;
            }
            Balance += amount;
            Console.WriteLine($"Deposited {amount:C} using cheque #{chequeNo}. New balance: {Balance:C}");
        }

        // Method to deposit a specified amount using a credit/debit card
        public void Deposit(double amount, string cardNo)
        {
            if (amount <= 0)
            {
                Console.WriteLine("Invalid amount. Deposit failed.");
                return;
            }
```

```
            if (string.IsNullOrWhiteSpace(cardNo) || cardNo.Length < 4)
            {
                Console.WriteLine("Invalid card number. Deposit failed.");
                return;
            }
            Balance += amount;
            Console.WriteLine($"Deposited {amount:C} using card ending in {cardNo[^4..]}. New balance:
{Balance:C}");
        }

        // Display balance
        public void DisplayBalance()
        {
            Console.WriteLine($"Current Balance: {Balance:C}");
        }

        // Main method for demonstration
        public static void Main(string[] args)
        {
            BankTransaction transaction = new BankTransaction();

            // Deposit in cash
            transaction.Deposit(500);
            transaction.DisplayBalance();

            // Deposit using a cheque
            transaction.Deposit(1000, 12345);
            transaction.DisplayBalance();

            // Deposit using a credit card
            transaction.Deposit(1500, "1234567890123456");
            transaction.DisplayBalance();
        }
    }
}
```

3. Write a program to create an interface Vehicle with methods Start() and Stop(). Implement this interface in a class Car.

```
using System;

namespace VehicleApp
{
  // Interface definition
  public interface Vehicle
  {
    void Start(); // Method to start the vehicle
    void Stop();  // Method to stop the vehicle
  }



  // Car class implementing the Vehicle interface
  public class Car : Vehicle
  {
    // Implementation of Start method
    public void Start()
    {
      Console.WriteLine("The car has started.");
    }

    // Implementation of Stop method
    public void Stop()
    {
      Console.WriteLine("The car has stopped.");
    }
  }

  // Main program to demonstrate the interface and class implementation
  public class Program
  {
    public static void Main(string[] args)
    {
      // Create a Car object
      Vehicle myCar = new Car();

      // Call the Start method
      myCar.Start();

      // Call the Stop method
      myCar.Stop();
    }
  }
}
```

4. **Write a program to create a dictionary of students and their grades. Perform the following operations:**
   - **Add a student and grade.**
   - **Remove a student by name.**
   - **Check if a student exists in the dictionary.**

```
using System;
using System.Collections.Generic;

namespace StudentGradeManagement
{
  public class Program
  {
    public static void Main(string[] args)
    {
      // Initialize a dictionary to store student names and their grades
      Dictionary<string, string> studentGrades = new Dictionary<string, string>();

      bool exit = false;
      while (!exit)
      {
        // Display menu options
        Console.WriteLine("\nStudent Grade Management System");
        Console.WriteLine("1. Add a Student and Grade");
        Console.WriteLine("2. Remove a Student by Name");
        Console.WriteLine("3. Check if a Student Exists");
        Console.WriteLine("4. Display All Students");
        Console.WriteLine("5. Exit");
        Console.Write("Choose an option: ");

        int choice = int.Parse(Console.ReadLine());
        if(choice <=0 || choice > 5)
        {
          Console.WriteLine("Invalid input. Please enter a valid option.");
          continue;
        }

        switch (choice)
        {
          case 1:
            // Add a student and grade
            Console.Write("Enter student's name: ");
            string studentName = Console.ReadLine();

            Console.Write("Enter student's grade: ");
            string grade = Console.ReadLine();

            // Add the student to the dictionary
            if (!studentGrades.ContainsKey(studentName))
            {
```

```
                studentGrades.Add(studentName, grade);
                Console.WriteLine($"Student {studentName} with grade {grade} added.");
            }
            else
            {
                Console.WriteLine("This student already exists in the dictionary.");
            }
            break;

        case 2:
            // Remove a student by name
            Console.Write("Enter the name of the student to remove: ");
            string removeName = Console.ReadLine();

            if (studentGrades.Remove(removeName))
            {
                Console.WriteLine($"Student {removeName} has been removed.");
            }
            else
            {
                Console.WriteLine("Student not found.");
            }
            break;

        case 3:
            // Check if a student exists
            Console.Write("Enter student's name to check: ");
            string checkName = Console.ReadLine();

            if (studentGrades.ContainsKey(checkName))
            {
                Console.WriteLine($"Student {checkName} exists with grade:
{studentGrades[checkName]}");
            }
            else
            {
                Console.WriteLine("Student not found.");
            }
            break;

        case 4:
            // Display all students and their grades
            Console.WriteLine("\nList of All Students:");
            foreach (var student in studentGrades)
            {
                Console.WriteLine($"{student.Key}: {student.Value}");
            }
            break;

        case 5:
            // Exit the program
            exit = true;
```

```
                Console.WriteLine("Exiting the program.");
                break;

            default:
                Console.WriteLine("Invalid option. Please try again.");
                break;
        }
      }
    }
  }
}
```

5. **Matrix Diagonal Sum**

Write a program to compute the sum of the primary diagonal elements of a square matrix. For example, for the matrix:

Ex:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

The output should be 1+5+9=15.

```
using System;

namespace MatrixDiagonalSum
{
  public class Program
  {
    public static void Main(string[] args)
    {
      // Get matrix dimensions from the user
      Console.Write("Enter the dimension of the square matrix (n x n): ");
      int n = int.Parse(Console.ReadLine());

      // Initialize the matrix
      int[,] matrix = new int[n, n];

      // Get matrix elements from the user
      Console.WriteLine("Enter the elements of the matrix:");
      for (int i = 0; i < n; i++)
      {
        for (int j = 0; j < n; j++)
        {
          Console.Write($"Element at position ({i+1},{j+1}): ");
          matrix[i, j] = int.Parse(Console.ReadLine());
        }
      }

      // Call the method to calculate the diagonal sum
      int diagonalSum = CalculateDiagonalSum(matrix, n);
```

```
            // Print the result
            Console.WriteLine($"The sum of the diagonal elements is: {diagonalSum}");
        }

        // Method to calculate the sum of the diagonal elements
        public static int CalculateDiagonalSum(int[,] matrix, int size)
        {
            int sum = 0;

            for (int i = 0; i < size; i++)
            {
                // Add the primary diagonal element
                sum += matrix[i, i]; // primary diagonal (top-left to bottom-right)
            }

            return sum;
        }
    }
}
```

6. Anagram Check

Write a program to determine if two given strings are anagrams of each other. For example, "listen" and "silent" are anagrams, but "hello" and "world" are not.

```
using System;

namespace AnagramCheck
{
    public class Program
    {
        public static void Main(string[] args)
        {
            // Input two strings
            Console.Write("Enter the first string: ");
            string str1 = Console.ReadLine().ToLower(); // Convert to lowercase for case-insensitive comparison

            Console.Write("Enter the second string: ");
            string str2 = Console.ReadLine().ToLower(); // Convert to lowercase for case-insensitive comparison

            // Call the method to check if they are anagrams
            bool isAnagram = AreAnagrams(str1, str2);

            // Print the result
            if (isAnagram)
            {
                Console.WriteLine($"'{str1}' and '{str2}' are anagrams.");
            }
            else
            {
```

```
            Console.WriteLine($"'{str1}' and '{str2}' are not anagrams.");
        }
    }

    // Method to check if two strings are anagrams
    public static bool AreAnagrams(string str1, string str2)
    {
        // If the strings are not of the same length, they cannot be anagrams
        if (str1.Length != str2.Length)
        {
            return false;
        }

        // Convert strings to character arrays and sort them
        char[] arr1 = str1.ToCharArray();
        char[] arr2 = str2.ToCharArray();

        Array.Sort(arr1); // Sort the first array
        Array.Sort(arr2); // Sort the second array

        // Compare the sorted arrays
        for (int i = 0; i < arr1.Length; i++)
        {
            if (arr1[i] != arr2[i])
            {
                return false; // If any character doesn't match, they are not anagrams
            }
        }

        return true; // If all characters match, they are anagrams
    }
  }
}
```

7. An Armstrong number of n :
n digits is a number such that the sum of its digits each raised to the power n where n equals the number itself. For example:
153 = 1^3 + 5^3 + 3^3
Write a program to check if a given number is an Armstrong number.

```
using System;

namespace ArmstrongNumber
{
  public class Program
  {
    public static void Main(string[] args)
    {
      // Input number from user
      Console.Write("Enter a number: ");
      int number = int.Parse(Console.ReadLine());

      // Call the method to check if it's an Armstrong number
      if (IsArmstrong(number))
      {
        Console.WriteLine($"{number} is an Armstrong number.");
      }
      else
      {
        Console.WriteLine($"{number} is not an Armstrong number.");
      }
    }
    // Method to check if a number is an Armstrong number
    public static bool IsArmstrong(int number)
    {
      int originalNumber = number;
      int sum = 0;
      int digits = number.ToString().Length; // Number of digits in the number

      // Loop through each digit, raise it to the power of the number of digits, and add to sum
      while (number > 0)
      {
        int digit = number % 10; // Get the last digit
        sum += (int)Math.Pow(digit, digits); // Raise the digit to the power of the number of digits and add
it to sum
        number /= 10; // Remove the last digit
      }
      // If the sum equals the original number, it's an Armstrong number
      return sum == originalNumber;
    }
  }
}
```