Set B Solution

1. WAP to get marks of 5 subjects and calculate percentage and grade accordingly.

```
using System;
class Program
{
    static void Main()
    {
        // Declare an array to hold marks for 5 subjects
        int[] marks = new int[5];
        int totalMarks = 0;
        double percentage;
        char grade;

        Console.WriteLine("Enter marks for 5 subjects (out of 100):");

        // Loop to get marks for each subject
        for (int i = 0; i < 5; i++)
        {
            Console.Write($"Subject {i + 1}: ");
            marks[i] = int.Parse(Console.ReadLine());

            // Validate marks input
            if (marks[i] < 0 || marks[i] > 100)
            {
                Console.WriteLine("Invalid input. Marks should be between 0 and 100.");
                i--; // Prompt for the input again
                continue;
            }
            totalMarks += marks[i];
        }

        // Calculate percentage
        percentage = Convert.ToDouble(totalMarks) / 5;

        // Determine grade based on percentage
        if (percentage >= 90)
        {
            grade = 'A';
        }
        else if (percentage >= 80)
        {
            grade = 'B';
        }
        else if (percentage >= 70)
        {
            grade = 'C';
        }
        else if (percentage >= 60)
        {
            grade = 'D';
        }
        else
        {
            grade = 'F';
        }

        // Display results
        Console.WriteLine("\nResults:");
        Console.WriteLine($"Total Marks: {totalMarks}/500");
        Console.WriteLine($"Percentage: {percentage:F2}%");
        Console.WriteLine($"Grade: {grade}");
    }
}
```

2. WAP to check whether the no. is Palindrome or not.

```
using System;
class Program
{
    static void Main()
    {
        Console.Write("Enter a number: ");
        int number = int.Parse(Console.ReadLine());
        int originalNumber = number; // Store the original number
        int reversedNumber = 0;

        // Reverse the number
        while (number > 0)
        {
            int remainder = number % 10;
            reversedNumber = (reversedNumber * 10) + remainder;
            number /= 10;
        }

        // Check if the original number and reversed number are the same
        if (originalNumber == reversedNumber)
        {
            Console.WriteLine($"{originalNumber} is a Palindrome.");
        }
        else
        {
            Console.WriteLine($"{originalNumber} is not a Palindrome.");
        }
    }
}
```

3. **WAP to implement class Device having properties DeviceID, DeviceName, AverageLife, OSType class Laptop having properties ModelName, LaptopType(eg: Office, Gaming, etc.) Inherit Device class in Laptop class and implement get and display details methods.**

```
using System;
class Device
{
    // Properties of Device class
    public int DeviceID { get; set; }
    public string DeviceName { get; set; }
    public int AverageLife { get; set; } // in years
    public string OSType { get; set; } // Operating System Type

    // Method to get details of the Device
    public virtual void GetDetails()
    {
        Console.Write("Enter Device ID: ");
        DeviceID = int.Parse(Console.ReadLine());
        Console.Write("Enter Device Name: ");
        DeviceName = Console.ReadLine();
        Console.Write("Enter Average Life (in years): ");
        AverageLife = int.Parse(Console.ReadLine());
        Console.Write("Enter OS Type: ");
        OSType = Console.ReadLine();
    }

    // Method to display details of the Device
    public virtual void DisplayDetails()
    {
        Console.WriteLine("\nDevice Details:");
        Console.WriteLine($"Device ID: {DeviceID}");
        Console.WriteLine($"Device Name: {DeviceName}");
        Console.WriteLine($"Average Life: {AverageLife} years");
        Console.WriteLine($"OS Type: {OSType}");
    }
}

class Laptop : Device
{
    // Properties specific to Laptop
    public string ModelName { get; set; }
    public string LaptopType { get; set; } // e.g., Office, Gaming, etc.

    // Method to get details of the Laptop
    public override void GetDetails()
    {
        // Get base class details
        base.GetDetails();

        // Get Laptop-specific details
        Console.Write("Enter Model Name: ");
        ModelName = Console.ReadLine();
```

```
            Console.Write("Enter Laptop Type (e.g., Office, Gaming): ");
            LaptopType = Console.ReadLine();
        }

        // Method to display details of the Laptop
        public override void DisplayDetails()
        {
            // Display base class details
            base.DisplayDetails();

            // Display Laptop-specific details
            Console.WriteLine($"Model Name: {ModelName}");
            Console.WriteLine($"Laptop Type: {LaptopType}");
        }
    }

    class Program
    {
        static void Main()
        {
            // Create an instance of Laptop
            Laptop myLaptop = new Laptop();

            // Get details of the Laptop
            Console.WriteLine("Enter details of the Laptop:");
            myLaptop.GetDetails();

            // Display details of the Laptop
            Console.WriteLine("\nDisplaying Laptop details:");
            myLaptop.DisplayDetails();
        }
    }
```

4. Create PhoneBook using Dictionary which stores ContactName as key and ContactNo. As value.

```
    using System;
    using System.Collections.Generic;
    class Program
    {
        static void Main()
        {
            Dictionary<string, string> contacts = new Dictionary<string, string>();
            bool running = true;

            while (running)
            {
                Console.WriteLine("\nPhoneBook Menu:");
                Console.WriteLine("1. Add Contact");
                Console.WriteLine("2. Display All Contacts");
                Console.WriteLine("3. Search Contact");
                Console.WriteLine("4. Delete Contact");
                Console.WriteLine("5. Exit");
                Console.Write("Enter your choice: ");
                int choice = int.Parse(Console.ReadLine());

                switch (choice)
                {
                  case 1:
                    Console.Write("Enter Contact Name: ");
                    string name = Console.ReadLine();
                    Console.Write("Enter Contact Number: ");
                    string number = Console.ReadLine();
                    contacts[name] = number;
                    break;
                  case 2:
                    if (contacts.Count == 0)
                    {
                      Console.WriteLine("PhoneBook is empty.");
                      break;
                    }
                    Console.WriteLine("\nPhoneBook Contacts:");
                    foreach (var contact in contacts)
                    {
                      Console.WriteLine($"Name: {contact.Key}, Number: {contact.Value}");
                    }
                    break;
                  case 3:
                    Console.Write("Enter Contact Name to Search: ");
                    name = Console.ReadLine();
                    if (contacts.ContainsKey(name))
                    {
                      Console.WriteLine($"Contact Found - Name: {name}, Number: {contacts[name]}");
                    }
                    else
                    {
                      Console.WriteLine($"Contact '{name}' not found.");
                    }
                    break;
                  case 4:
                    Console.Write("Enter Contact Name to Delete: ");
                    name = Console.ReadLine();
                    if (contacts.Remove(name))
                    {
                      Console.WriteLine($"Contact '{name}' deleted successfully.");
                    }
                    else
                    {
                      Console.WriteLine($"Contact '{name}' not found.");
                    }
                    break;
                  case 5:
                    running = false;
                    Console.WriteLine("Exiting PhoneBook. ");
```

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

5. **WAP to demonstrate Error handling, which takes 2 string as input and throws exception when two strings are not equal.**

```
using System;

class StringsNotEqualException : Exception
{
    public StringsNotEqualException(string message) : base(message) { }
}

class Program
{
    static void Main()
    {
        try
        {
            // Input two strings
            Console.Write("Enter the first string: ");
            string str1 = Console.ReadLine().ToLower();

            Console.Write("Enter the second string: ");
            string str2 = Console.ReadLine().ToLower();

            // Check if the strings are equal
            if (!str1.Equals(str2))
            {
                throw new StringsNotEqualException("The two strings are not equal.");
            }

            Console.WriteLine("The two strings are equal!");
        }
        catch (StringsNotEqualException ex)
        {
            Console.WriteLine($"Exception: {ex.Message}");
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Unexpected Error: {ex.Message}");
        }
    }
}
```

6. **WAP which has class Project with properties ProjectName, LineOfCode, FrontendTechnology, BackendTechnology and Get and Display methods. Create List of objects of Project class and demonstrate Add and Remove operations of List.**

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

class Project
{
    // Properties of the Project class
    public string ProjectName { get; set; }
    public int LineOfCode { get; set; }
    public string FrontendTechnology { get; set; }
    public string BackendTechnology { get; set; }

    // Method to get project details from the user
    public void GetDetails()
    {
        Console.Write("Enter Project Name: ");
        ProjectName = Console.ReadLine();

        Console.Write("Enter Line of Code: ");
        LineOfCode = int.Parse(Console.ReadLine());

        Console.Write("Enter Frontend Technology: ");
        FrontendTechnology = Console.ReadLine();

        Console.Write("Enter Backend Technology: ");
        BackendTechnology = Console.ReadLine();
    }

    // Method to display project details
    public void DisplayDetails()
    {
        Console.WriteLine($"\nProject Name: {ProjectName}");
        Console.WriteLine($"Line of Code: {LineOfCode}");
        Console.WriteLine($"Frontend Technology: {FrontendTechnology}");
        Console.WriteLine($"Backend Technology: {BackendTechnology}");
    }
}

class Program
{
    static void Main()
    {
        // List to hold Project objects
        List<Project> projectList = new List<Project>();

        bool running = true;
        while (running)
        {
            Console.WriteLine("\nMenu:");
            Console.WriteLine("1. Add a Project");
            Console.WriteLine("2. Display All Projects");
            Console.WriteLine("3. Remove a Project");
            Console.WriteLine("4. Exit");
            Console.Write("Enter your choice: ");
            int choice = int.Parse(Console.ReadLine());

            switch (choice)
            {
                case 1: // Add a Project
                    Project newProject = new Project();
                    Console.WriteLine("Enter details for the new project:");
                    newProject.GetDetails();
                    projectList.Add(newProject);
                    break;

                case 2: // Display All Projects
                    if (projectList.Count == 0)
                    {
                        Console.WriteLine("No projects to display.");
                    }
                    else
                    {
                        Console.WriteLine("\nProjects in the List:");
                        foreach (var project in projectList)
                        {
                            project.DisplayDetails();
                        }
                    }
                    break;

                case 3: // Remove a Project
                    Console.Write("Enter the No. of Project to remove: ");
                    int index = int.Parse(Console.ReadLine());
                    projectList.RemoveAt(index-1);
                    break;

                case 4: // Exit
                    running = false;
                    Console.WriteLine("Exiting program. Goodbye!");
                    break;

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

7. WAP to convert decimal no. to binary using Recurrsion.

```
using System;
class Program
{
  // Recursive method to convert decimal to binary
  static string DecimalToBinary(int n)
  {
    // Base case: if the number is 0, return an empty string
    if (n == 0)
    {
      return "";
    }

    // Recursively divide the number by 2 and append the remainder (0 or 1)
    return DecimalToBinary(n / 2) + (n % 2).ToString();
  }

  static void Main()
  {
    Console.Write("Enter a decimal number: ");
    int decimalNumber = int.Parse(Console.ReadLine());

    // Special case for 0, since the recursion will return an empty string
    if (decimalNumber == 0)
    {
      Console.WriteLine("Binary: 0");
    }
    else
    {
      // Call the recursive method and display the binary representation
      string binaryRepresentation = DecimalToBinary(decimalNumber);
      Console.WriteLine($"Binary: {binaryRepresentation}");
    }
  }
}
```