Department of Statistics & Computer Science
University of Kelaniya
Academic Year (23/24)
COSC 31112 /BECS 31242
Visual Programming

Practical Guide 06

1. Create a Windows Form with buttons and textboxes to add numbers using overloaded methods.
   (i) Add buttons, labels and a textbox and design the interface as given below.
   (ii) Create a new class file Calculator.cs and write the below code

```
namespace WinFormsApp3
{
    2 references
    public class Calculator
    {
        // Add two integers
        1 reference
        public int Add(int a, int b)
        {
            return a + b;
        }

        // Add three integers (overloaded method)
        1 reference
        public int Add(int a, int b, int c)
        {
            return a + b + c;
        }

        // Add two doubles (overloaded method)
        0 references
        public double Add(double a, double b)
        {
            return a + b;
        }
    }
```

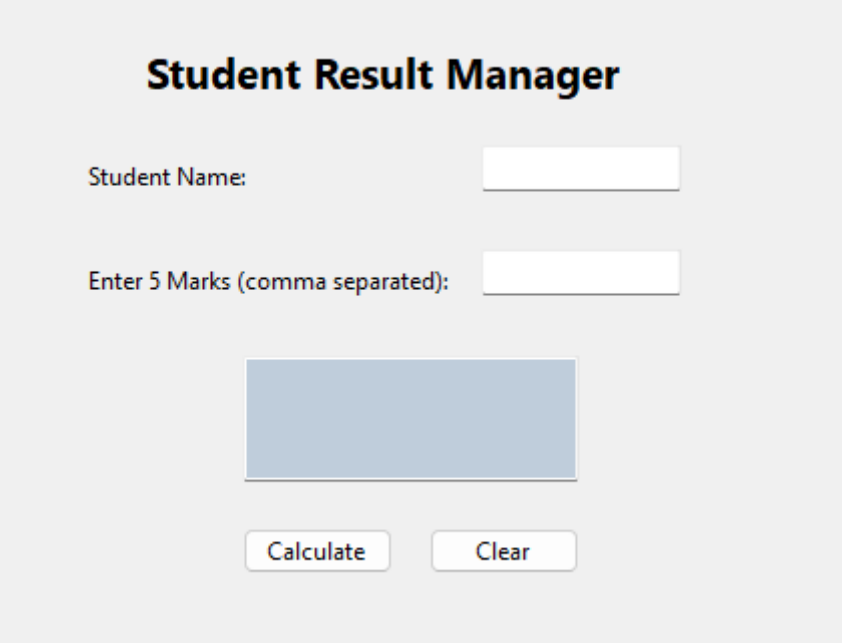
   (iii) Double click the button to create btnAdd_Click event and write the below code.

```
1 reference
private void btnAdd_Click(object sender, EventArgs e)
{
    Calculator calc = new Calculator();

    int num1, num2, num3;
    bool isNum3Empty = string.IsNullOrEmpty(txtNum3.Text);

    if (int.TryParse(txtNum1.Text, out num1) && int.TryParse(txtNum2.Text, out num2))
    {
        if (isNum3Empty)
        {
            // Use Add(int, int)
            int result = calc.Add(num1, num2);
            lblResult.Text = $"Sum of two numbers: {result}";
        }
        else if (int.TryParse(txtNum3.Text, out num3))
        {
            // Use Add(int, int, int)
            int result = calc.Add(num1, num2, num3);
            lblResult.Text = $"Sum of three numbers: {result}";
        }
        else
        {
            lblResult.Text = "Invalid input in third number.";
        }
    }
    else
    {
        lblResult.Text = "Invalid input in first two numbers.";
    }
```

(iv) Run the application by pressing F5

2. Create a Windows Forms Application to manage student exam results with features to add student marks, calculate average, find max marks, assign grades, and display results.

   (i) Add buttons, labels and a textbox and design the interface as given below. Set one textbox to be multiline and read-only.



   (ii) Add a new class called Student in your project.
   - Use private fields and public properties for encapsulation (access modifiers).
   - Add methods for:
     - Calculating average
     - Finding the max mark
     - Getting grade based on average
     - Recursively calculate factorial for a mark (just to show recursion)

```
namespace WinFormsApp3
{
    3 references
    public class Student
    {

        private string name;
        private int[] marks;

        1 reference
        public Student(string name, int[] marks)
        {
            this.name = name;
            this.marks = marks;
        }

        2 references
        public double CalculateAverage()
        {
            int total = 0;
            foreach (int mark in marks)
            {
                total += mark;
            }
            return (double)total / marks.Length;
        }

        0 references
        public string GetGrade()
        {
            double avg = CalculateAverage();
            if (avg >= 75) return "Distinction";
            else if (avg >= 60) return "Credit";
            else if (avg >= 50) return "Pass";
            else return "Fail";
        }

        1 reference
        public string GetGrade(double avg)
        {
            if (avg >= 75) return "Distinction (External)";
            else if (avg >= 60) return "Credit (External)";
            else if (avg >= 50) return "Pass (External)";
            else return "Fail (External)";
        }

        2 references
        public int FindMaxRecursive(int[] arr, int n)
        {
            if (n == 1) return arr[0];
            return Math.Max(arr[n - 1], FindMaxRecursive(arr, n - 1));
        }
```

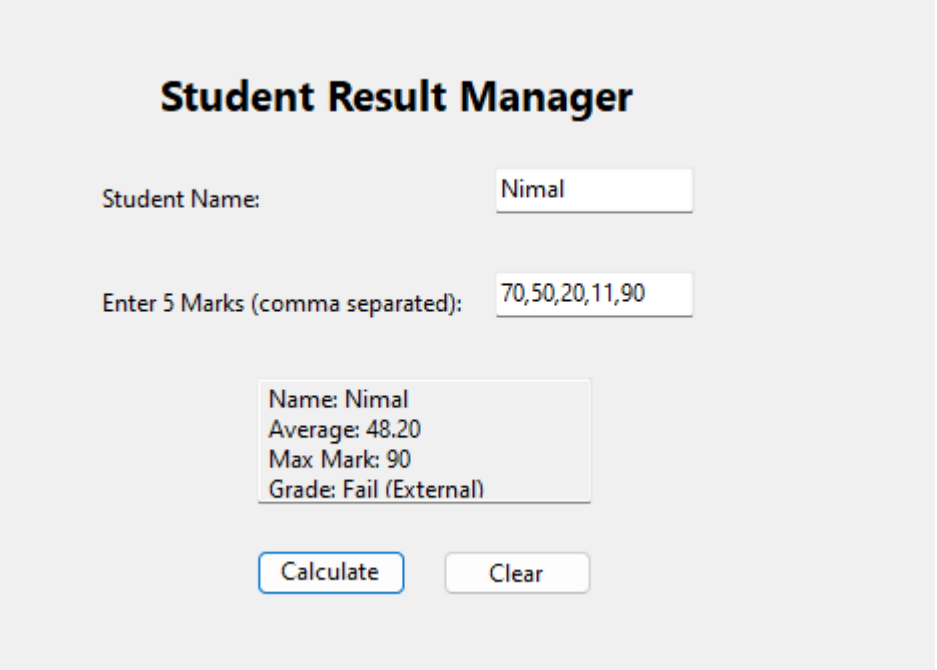
```
        1 reference
        public string GetSummary()
        {
            double avg = CalculateAverage();
            int maxMark = FindMaxRecursive(marks, marks.Length);
            string grade = GetGrade(avg);
            return $"Name: {name}\r\nAverage: {avg:F2}\r\nMax Mark: {maxMark}\r\nGrade: {grade}";
        }
    }

}
```

(iii) Double click on "calculate" button and write the below code.

```
        1 reference
        private void btnCalculate_Click(object sender, EventArgs e)
        {
            try
            {
                string studentName = txtName.Text;
                string[] parts = txtMarks.Text.Split(',');
                int[] marks = Array.ConvertAll(parts, int.Parse);

                if (marks.Length != 5)
                {
                    MessageBox.Show("Please enter exactly 5 marks.");
                    return;
                }

                Student s = new Student(studentName, marks);
                txtResult.Text = s.GetSummary();
            }
            catch (Exception ex)
            {
                MessageBox.Show("Error: " + ex.Message);
            }
        }
```

(iv) Double click on "calculate" button and write the below code

```
        1 reference
        private void btnClear_Click(object sender, EventArgs e)
        {
            txtName.Clear();
            txtMarks.Clear();
            txtResult.Clear();
        }
    }
}
```



(v) Run the application by pressing F5