7. Build a Windows Form application that performs arithmetic operations on two numbers. Use Textbox controls to input and display the numbers, Label controls to describe each field, and Button controls to perform the arithmetic operations. Use a Combo Box control to select the operator (+, -, *, /). Use an Array class to store the history of the arithmetic operations performed. Add a menu to the form with options to clear the history and exit the application.

Steps:
C# Windows Forms application that does exactly what you asked: takes two numbers, lets you pick an operator from a **ComboBox**, computes the result when you click **Calculate**, stores each calculation in a **history array** (using System.Array helpers), shows the history in a ListBox, and has a **menu** with *Clear History* and *Exit*.

Code — single file (Program.cs):

```
using System;
using System.Windows.Forms;

namespace CalcHistoryApp
{
   static class Program
   {
      [STAThread]
      static void Main()
      {
         Application.EnableVisualStyles();
         Application.SetCompatibleTextRenderingDefault(false);
         Application.Run(new MainForm());
      }
   }

   public class MainForm : Form
   {
      // UI controls
      private Label lblNum1, lblNum2, lblOperator, lblResult, lblHistory;
      private TextBox txtNum1, txtNum2, txtResult;
      private ComboBox cmbOperator;
      private Button btnCalculate, btnClearInputs;
      private ListBox lstHistory;
      private MenuStrip menuStrip;
      private ToolStripMenuItem fileMenu, clearHistoryMenuItem, exitMenuItem;

      // History stored using an Array (string array)
      private string[] history = new string[10]; // initial capacity
      private int histCount = 0; // number of entries stored

      public MainForm()
      {
         InitializeComponent();
      }

      private void InitializeComponent()
      {
         // Form
         this.Text = "Arithmetic Calculator with History";
         this.StartPosition = FormStartPosition.CenterScreen;
         this.ClientSize = new System.Drawing.Size(600, 380);
         this.FormBorderStyle = FormBorderStyle.FixedDialog;
         this.MaximizeBox = false;

         // Menu
```

```
        menuStrip = new MenuStrip();
        fileMenu = new ToolStripMenuItem("File");
        clearHistoryMenuItem = new ToolStripMenuItem("Clear History");
        exitMenuItem = new ToolStripMenuItem("Exit");

        clearHistoryMenuItem.Click += ClearHistoryMenuItem_Click;
        exitMenuItem.Click += ExitMenuItem_Click;

        fileMenu.DropDownItems.Add(clearHistoryMenuItem);
        fileMenu.DropDownItems.Add(new ToolStripSeparator());
        fileMenu.DropDownItems.Add(exitMenuItem);
        menuStrip.Items.Add(fileMenu);
        this.MainMenuStrip = menuStrip;
        this.Controls.Add(menuStrip);

        // Labels and TextBoxes for numbers
        lblNum1 = new Label() { Text = "Number 1:", Left = 20, Top = 40, Width = 70 };
        txtNum1 = new TextBox() { Left = 100, Top = 36, Width = 140 };

        lblNum2 = new Label() { Text = "Number 2:", Left = 20, Top = 80, Width = 70 };
        txtNum2 = new TextBox() { Left = 100, Top = 76, Width = 140 };

        // Operator ComboBox
        lblOperator = new Label() { Text = "Operator:", Left = 260, Top = 40, Width = 70 };
        cmbOperator = new ComboBox() { Left = 335, Top = 36, Width = 80, DropDownStyle =
ComboBoxStyle.DropDownList };
        cmbOperator.Items.AddRange(new object[] { "+", "-", "*", "/" });
        cmbOperator.SelectedIndex = 0;

        // Calculate button
        btnCalculate = new Button() { Text = "Calculate", Left = 260, Top = 72, Width = 120, Height = 28 };
        btnCalculate.Click += BtnCalculate_Click;
        this.AcceptButton = btnCalculate; // Enter triggers calculation

        // Clear Inputs button
        btnClearInputs = new Button() { Text = "Clear Inputs", Left = 390, Top = 72, Width = 120, Height = 28 };
        btnClearInputs.Click += BtnClearInputs_Click;

        // Result label & textbox
        lblResult = new Label() { Text = "Result:", Left = 20, Top = 120, Width = 70 };
        txtResult = new TextBox() { Left = 100, Top = 116, Width = 140, ReadOnly = true };

        // History label & ListBox
        lblHistory = new Label() { Text = "History:", Left = 20, Top = 160, Width = 70 };
        lstHistory = new ListBox() { Left = 20, Top = 185, Width = 540, Height = 160 };

        // Add controls to form
        this.Controls.Add(lblNum1);
        this.Controls.Add(txtNum1);
        this.Controls.Add(lblNum2);
        this.Controls.Add(txtNum2);
        this.Controls.Add(lblOperator);
        this.Controls.Add(cmbOperator);
        this.Controls.Add(btnCalculate);
        this.Controls.Add(btnClearInputs);
        this.Controls.Add(lblResult);
        this.Controls.Add(txtResult);
        this.Controls.Add(lblHistory);
        this.Controls.Add(lstHistory);
```

```
    }

    private void BtnCalculate_Click(object sender, EventArgs e)
    {
        // Parse inputs
        if (!double.TryParse(txtNum1.Text.Trim(), out double a))
        {
            MessageBox.Show("Please enter a valid number for Number 1.", "Input Error",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
            txtNum1.Focus();
            return;
        }

        if (!double.TryParse(txtNum2.Text.Trim(), out double b))
        {
            MessageBox.Show("Please enter a valid number for Number 2.", "Input Error",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
            txtNum2.Focus();
            return;
        }

        string op = cmbOperator.SelectedItem?.ToString() ?? "+";
        double result = 0.0;
        bool ok = true;

        switch (op)
        {
            case "+":
                result = a + b;
                break;
            case "-":
                result = a - b;
                break;
            case "*":
                result = a * b;
                break;
            case "/":
                if (b == 0)
                {
                    MessageBox.Show("Division by zero is not allowed.", "Math Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                    ok = false;
                }
                else
                {
                    result = a / b;
                }
                break;
            default:
                MessageBox.Show("Unknown operator selected.", "Operator Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                ok = false;
                break;
        }

        if (!ok) return;

        // Show result
        txtResult.Text = result.ToString();
```

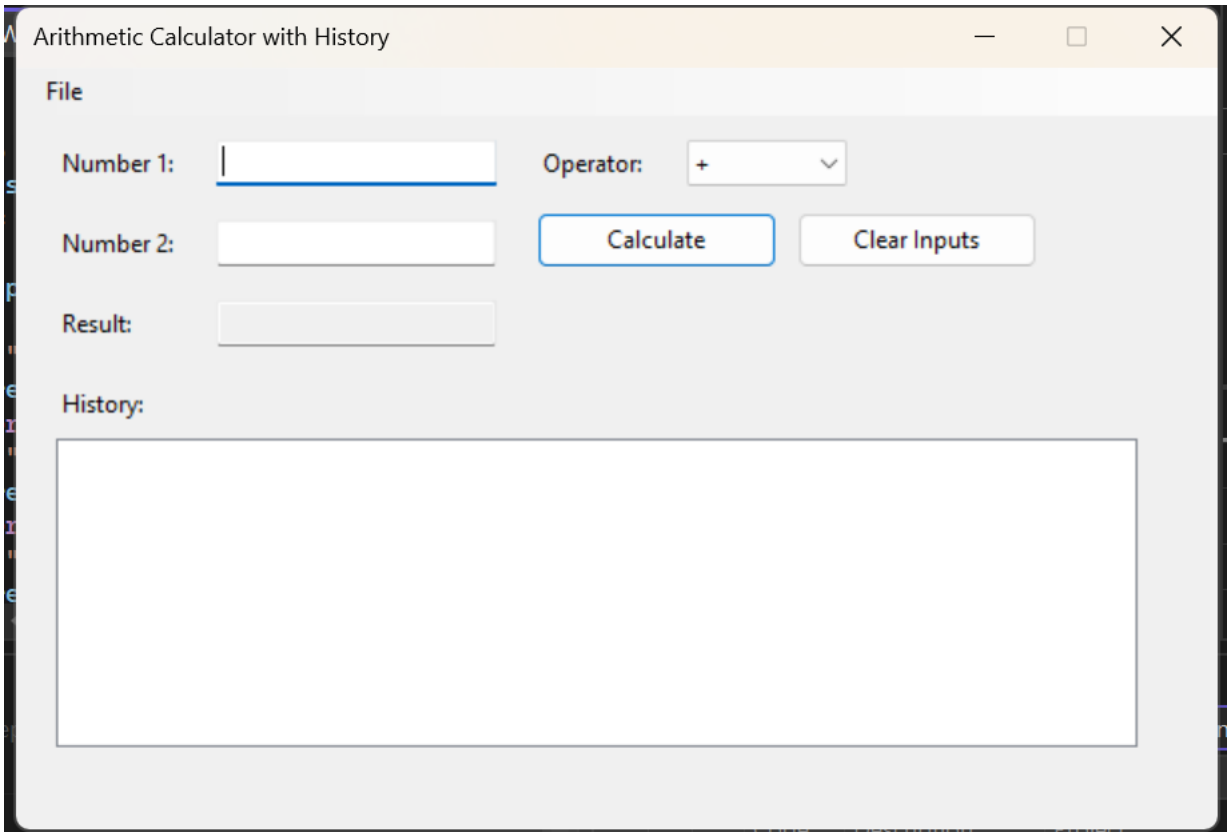
```
            // Create history entry and add to array & listbox
            string entry = $"{a} {op} {b} = {result}";
            AddToHistory(entry);
        }

        private void AddToHistory(string entry)
        {
            // If the array is full, resize it using Array.Resize (uses System.Array functionality)
            if (histCount >= history.Length)
            {
                // Double capacity
                Array.Resize(ref history, history.Length * 2);
            }

            history[histCount++] = entry;
            lstHistory.Items.Add(entry);
        }

        private void ClearHistoryMenuItem_Click(object sender, EventArgs e)
        {
            // Clear the array content and reset the counter
            Array.Clear(history, 0, history.Length); // clears all elements to null
            history = new string[10];               // reset to default size
            histCount = 0;
            lstHistory.Items.Clear();
        }

        private void ExitMenuItem_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void BtnClearInputs_Click(object sender, EventArgs e)
        {
            txtNum1.Clear();
            txtNum2.Clear();
            txtResult.Clear();
            cmbOperator.SelectedIndex = 0;
            txtNum1.Focus();
        }
    }
}
```

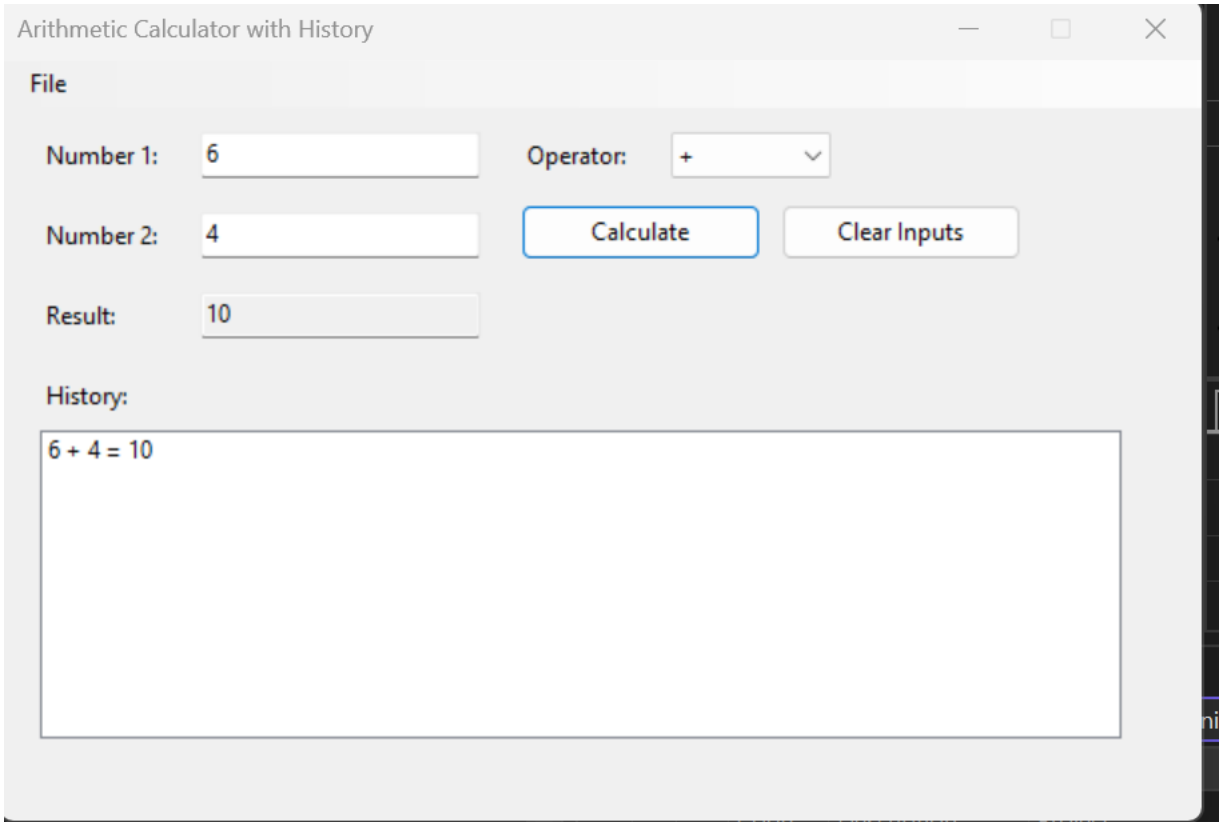
Step-by-step explanation (what each part does)

1. **Project setup**
    - Create a new **Windows Forms App** project in Visual Studio (C#).
    - Replace the default Program.cs with the single-file code above (or add MainForm as a new form and paste code accordingly).
    - Build and run (F5).
2. **Program entry**
    - Main() sets up the Windows Forms environment and opens MainForm.
3. **UI layout (inside InitializeComponent)**
    - MenuStrip with a File menu containing:
        - **Clear History** — clears the stored history and the ListBox display.
        - **Exit** — closes the app.
    - Label + TextBox for **Number 1** and **Number 2**.
    - ComboBox (cmbOperator) for operator selection + - * /. DropDownStyle set to DropDownList so user must pick from the list.

- Button **Calculate** triggers the computation.
- Button **Clear Inputs** resets the number fields and result.
- TextBox txtResult displays the computed result (read-only).
- ListBox lstHistory shows the textual history of past calculations.

4. **Parsing inputs & validation**
   - double.TryParse validates numeric input and shows a warning if input is invalid.
   - For division, the code checks for divide-by-zero and shows an error message.
5. **Performing the operation**
   - Uses a switch on the operator selected in the ComboBox.
   - For each operator it computes result and stores it in txtResult.
6. **History storage using an *Array***
   - The history variable is declared as a string[] and used to store textual descriptions of each operation ("5 + 3 = 8").
   - When histCount reaches the current array length, Array.Resize(ref history, newSize) doubles capacity. This is a direct use of the Array class helper.
   - To clear history, Array.Clear(history, 0, history.Length) is used to zero-out the array, then we reinitialize it to a default size. This demonstrates Array.Clear as a System.Array operation.
7. **UI history display**
   - Each new history entry is also added to lstHistory.Items so the user sees the chronological list.
8. **Good UX touches**
   - this.AcceptButton = btnCalculate; lets pressing **Enter** trigger Calculate.
   - txtResult is read-only to prevent accidental edits.
   - Form is centered and fixed-size for a simple, predictable layout.




Arithmetic Calculator with History
File
Number 1:
6
Operator:
+
Number 2:
4
Calculate
Clear Inputs
Result:
10
History:
6 + 4 = 10