Assignment #2

Type	Assignment
Created	@February 2, 2024 10:42 AM

👤 **Names:** Dia Villalpando, Ana Di Tano, Karla Vazquez, SongYu Chen, Addison Baum

🔗 **Repo:** github.com/diavillalpando/CS-392-Assignments

📖 Contents

📅 **Due Date(s): Feb 6, 2023**

Part 0:

1. **Watch/Review the following videos/links**
    - Namespaces: Video
    - Properties: Link, Link
2. **In the textbook:**
    a. Read 1.8, pp. 27-41; Complete Short Answer Problems on p. 48: 12, 13

Submission:

12. Figure 1-49 shows the Visual Studio IDE. What are the names of the four areas that are indicated in the figure?

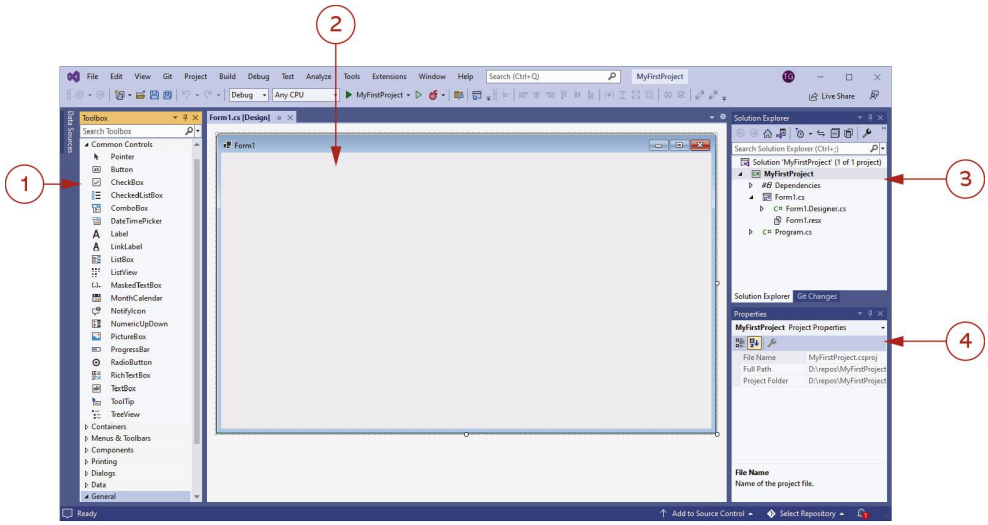


Figure 1-49 **The Visual Studio IDE**

1. Toolbox
2. Designer Window
3. Solution Explorer Window
4. Properties Window

13. What is the purpose of the *Toolbox* in the Visual Studio environment?

    a. It is the window that allows you to select the controls you want to use in the application's user interface

b. Read Ch. 2; Complete Programming Problems, 1, 4, 5

1. Look at the following list of Latin words and their meanings:

Latin	English
autem	up
descendit	down
medium	center

Create an application that translates the Latin words to English. The form should have three buttons, one for each Latin word. When the user clicks a button, the application should display

the English translation in a Label control.

📌 **Submission:**

4. **Joke and Punch Line:** A joke typically has two parts: a setup and a punch line. For example, this might be the setup for a joke:

   How many programmers does it take to change a lightbulb?

   And this is the punch line:

   None. That's a hardware problem.

   Think of your favorite joke and identify its setup and punch line. Then, create an application that has a Label and two buttons on a form. One of the buttons should read "Setup" and the other button should read "Punch line." When the *Setup* button is clicked, display the joke's setup in the Label control. When the *Punch line* button is clicked, display the joke's punch line in the Label control.

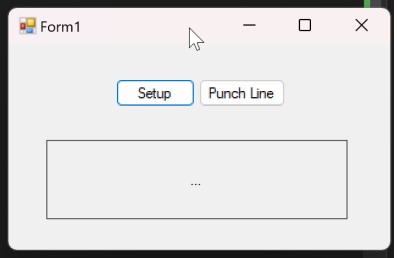
📌 **Submission:**

```
namespace part0_2_B_4
{
    3 references
    public partial class Form1 : Form
    {
        private string setup = "What's the advantage of living in Switzerland?";
        private string punch = "Well, the flag is a big plus!";
        1 reference
        public Form1()
        {
            InitializeComponent();
        }

        1 reference
        private void sayPunchBtn_Click(object sender, EventArgs e)
        {
            jokeLabel.Text = punch;
        }

        1 reference
        private void SaySetupBtn_Click(object sender, EventArgs e)
        {
            jokeLabel.Text = setup;
        }

    }
}
```



5. **Heads or Tails:** In the Student Sample Programs that accompany this book you will find a folder named *Images\Coins* that contains images showing the heads and tails sides of a coin. Create an application with a *Show Heads* button and a *Show Tails* button. When the user clicks the *Show Heads* button, an image of the heads side of a coin should appear. When the user clicks the *Show Tails* button, an image of the tails side of a coin should appear. Figure 2-82 shows examples of how the application's form might appear.

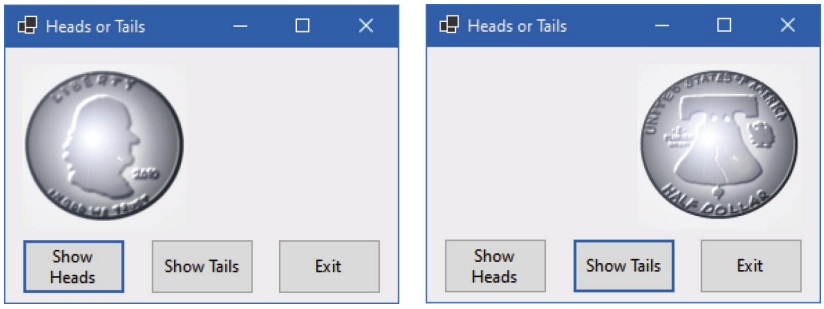
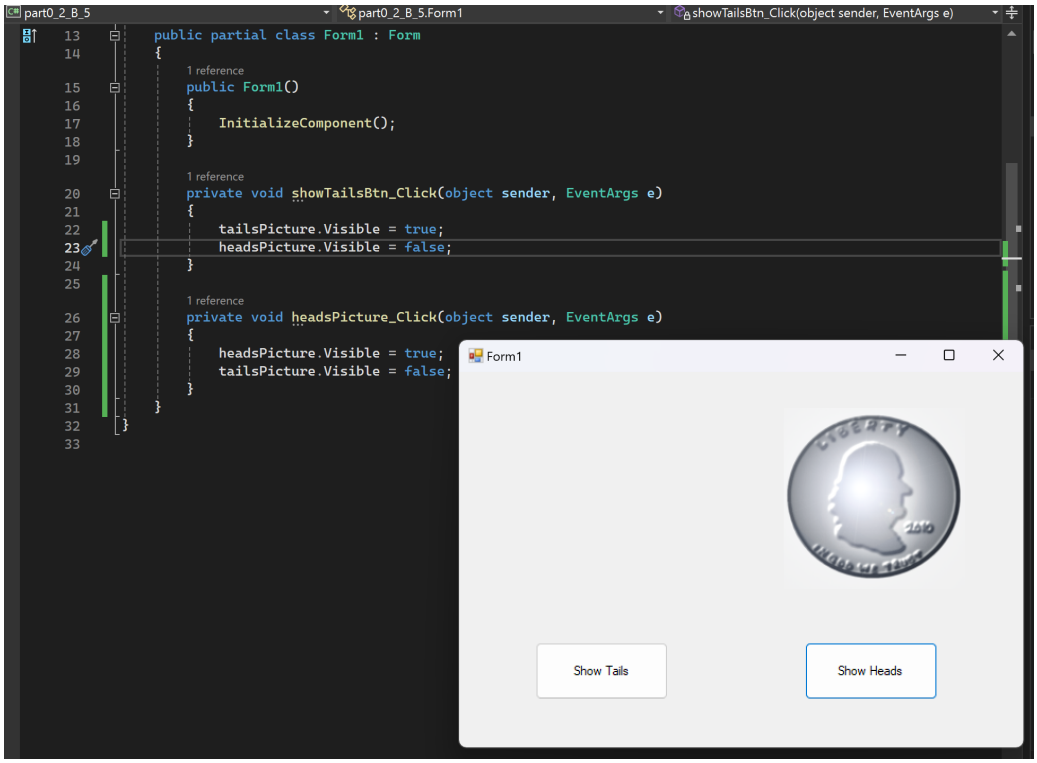


Figure 2-82 **The *Heads or Tails* application**

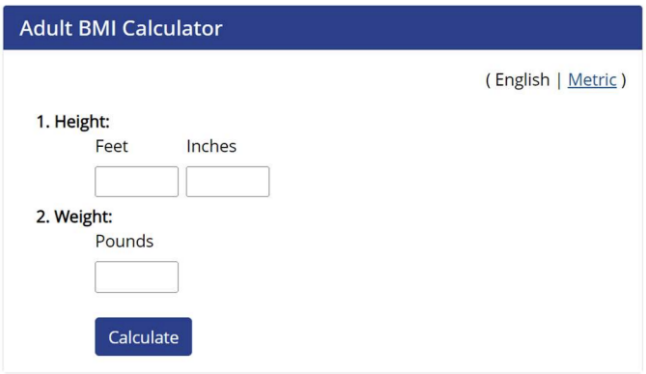
📌 **Submission:**



c. Pre-read Ch. 3 for next week

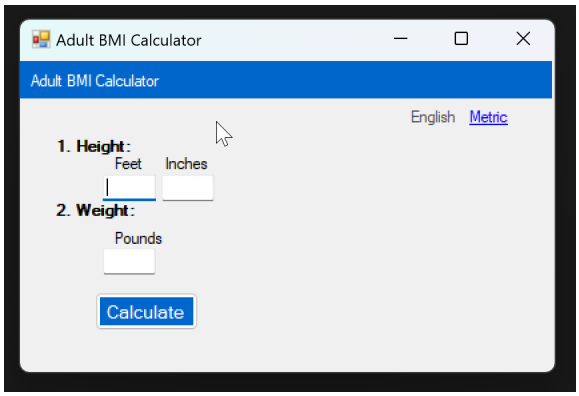
Part 1:

a. Implement a BMI Calculator just like the one the CDC uses at this link



Implement it as closely as possible to the screenshot above, including the toggle for *(English | Metric)*

📌 **Submission:**



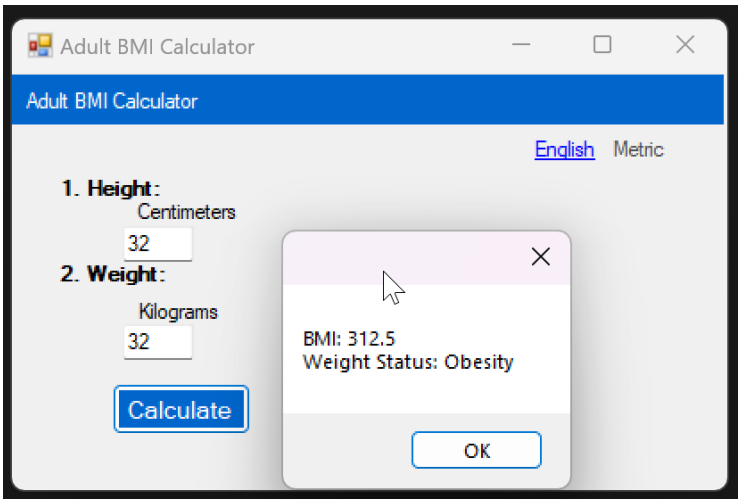
b. Modify the Calculate button so that a message box is displayed with the following text strings: `Underweight`, `Healthy Weight`, `Overweight`, `Obesity`, based on the table below.

BMI	Weight Status
Below 18.5	Underweight
18.5 – 24.9	Healthy Weight
25.0 – 29.9	Overweight
30.0 and Above	Obesity

Ref: https://www.cdc.gov/healthyweight/assessing/bmi/adult_bmi/index.html#Interpreted

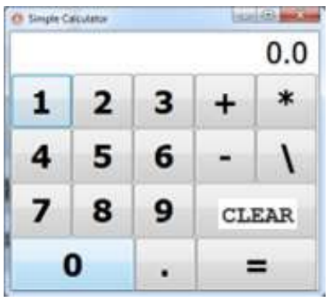
So for example, if the BMI calculated is 26.0, the word “Overweight” would appear in a message box.

📌 **Submission:**



Part 2

Implement a calculator App. The GUI should look very similar to the one shown below. It of course should work like a real calculator. Clicking on the numbers should update Text Box at the top. Users should also be able to type numbers directly in the Text Box. Don't worry about handling exceptions.



📌 **Submission:** Demo: 3 * 12