2019 年 5 月 23 日（木）実施

式と演算子

式

式とは，1 つの値，オブジェクト，メソッド，または名前空間(*)として評価することの出来る，1 つ以上のオペランドと，0 個以上の演算子の並びである。 式には，リテラル値(**)，メソッドの呼び出し，演算子とそのオペランド，または簡易名を含めることが出来る。 単純な名前には，変数，型メンバー，メソッドパラメーター，名前空間，または型の名前を指定出来る。

(*)**名前空間**　その内部にある識別子（型，関数，変数などの名前）のスコープ（適用範囲）を定める宣言領域で，コードを論理グループにまとめたり，名前の競合を回避したりするために使用される。

(**)**リテラル値**　数値や文字列のデータ値で，すべてのリテラル値には型が関連付けられている。

演算子

演算子とは，式または文の中で 1 つ以上のオペランドに適用されるプログラム要素である。各演算子には優先順位が定義されている。 優先順位のレベルが異なる複数の演算子を含む式の場合，演算子の優先順位によって演算子が評価される順序が決定される。C#言語の主な演算子を優先順位の順に挙げる（赤字が演算子，黒字がオペランドを表す）。各分類の中の優先順位は等しい。

分類	表現	説明
	x.y	メンバーアクセス
	f(x)	メソッドの呼び出し
	a[x]	配列アクセス，インデクサーアクセス
	x++	後置インクリメント（x=x+1）x++の参照は x の値となる。
	x−−	後置デクリメント（x=x−1）
	new T(...)	オブジェクトの作成
単項演算子	!x	論理否定
	++x	前置インクリメント　++x の参照は x+1 の値となる。
	−−x	前置デクリメント
	(T)x	キャスト　x を明示的に T 型に変換する。
乗法演算子	x*y	乗算
	x/y	除算
	x%y	剰余算
加法演算子	x+y	加算，文字列の連結
	x−y	減算

＊ 括弧を使用すると，演算子の優先順位を変更することが出来，括弧内を優先して評価する。

＊＊ 更に優先順位の低い，**関係演算子**，**等値演算子**，**論理演算子**については，次回に取り上げる。

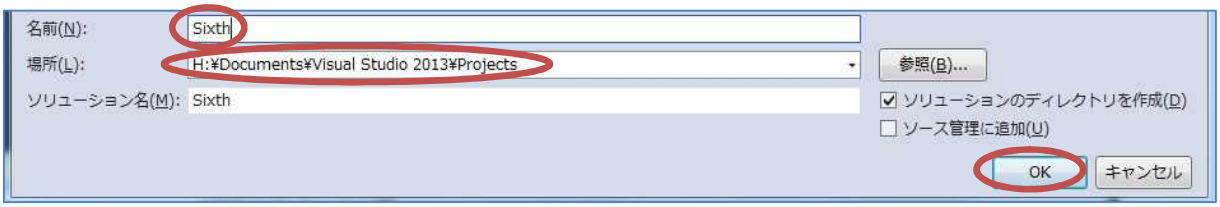
本日の課題

複数のフォームを利用して，平均値及び BMI を計算するプログラムを作成する。

手順

1）プロジェクトの作成

Visual Studio 2013 を起動したら，［ファイル］→［新規作成］→［プロジェクト］と辿って，プロジェクトを作成する。『新しいプロジェクト』ダイアログボックスでは，プログラミング言語を『Visual C#』，プロジェクトテンプレートとしては，『Windows フォームアプリケーション』を選択し，『名前』を「Sixth」に書き換え，『場所』が「H:¥Documents¥Visual Studio 2013¥Projects」となっていることを確認してから『OK』を押す（詳細は第 1 回の教材を参照）。



2）コントロールの配置及びフォームの作成

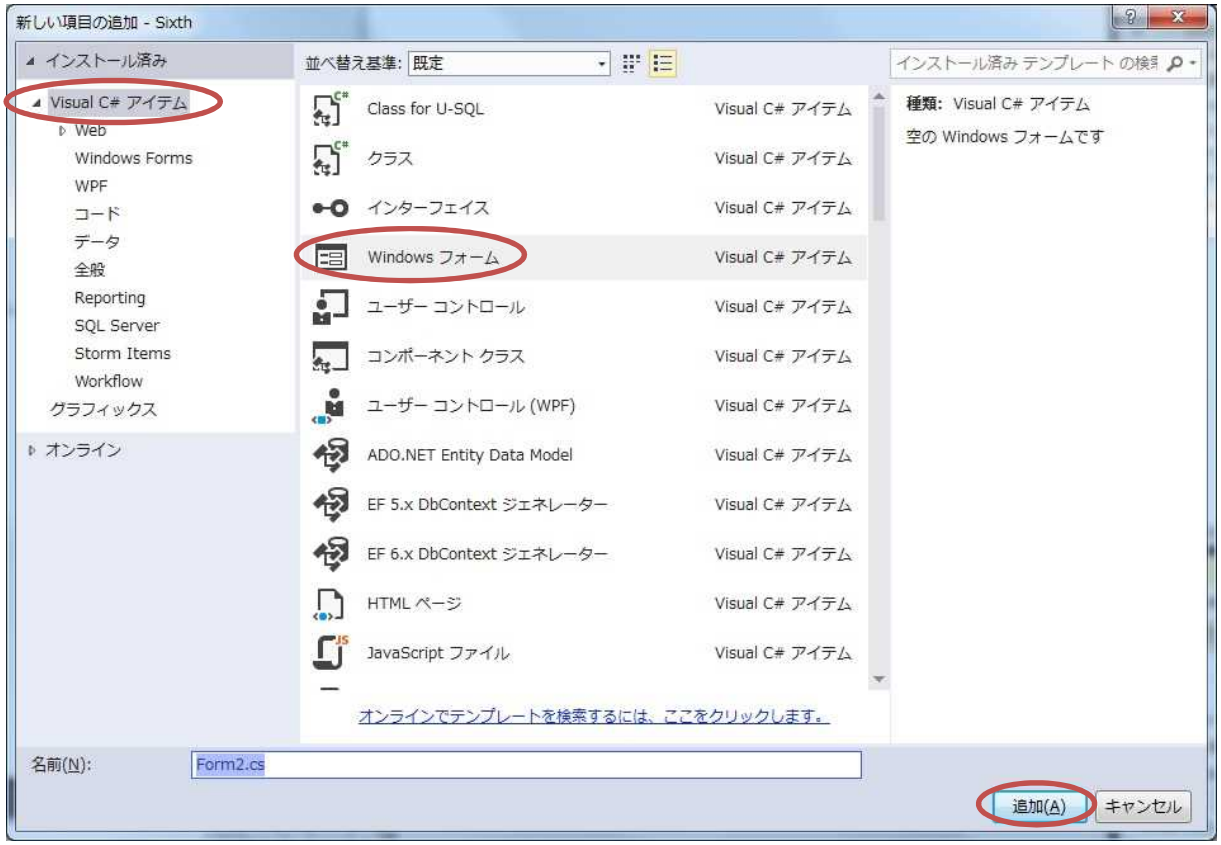
今後，フォーム上に配置するコントロールのプロパティのフォントサイズは全て 14 ポイントに変更するものとする。

Form1 上にボタンを 2 つ貼り付ける。それぞれのプロパティは次の様に設定する。

【Form1】Text　「**ポータル**」
　　Icon　自作のアイコン
【button1】Text　「**平均値の計算**」
【button2】Text　「**BMI の計算**」

[プロジェクト] → [Windows フォームの追加]を選択する。

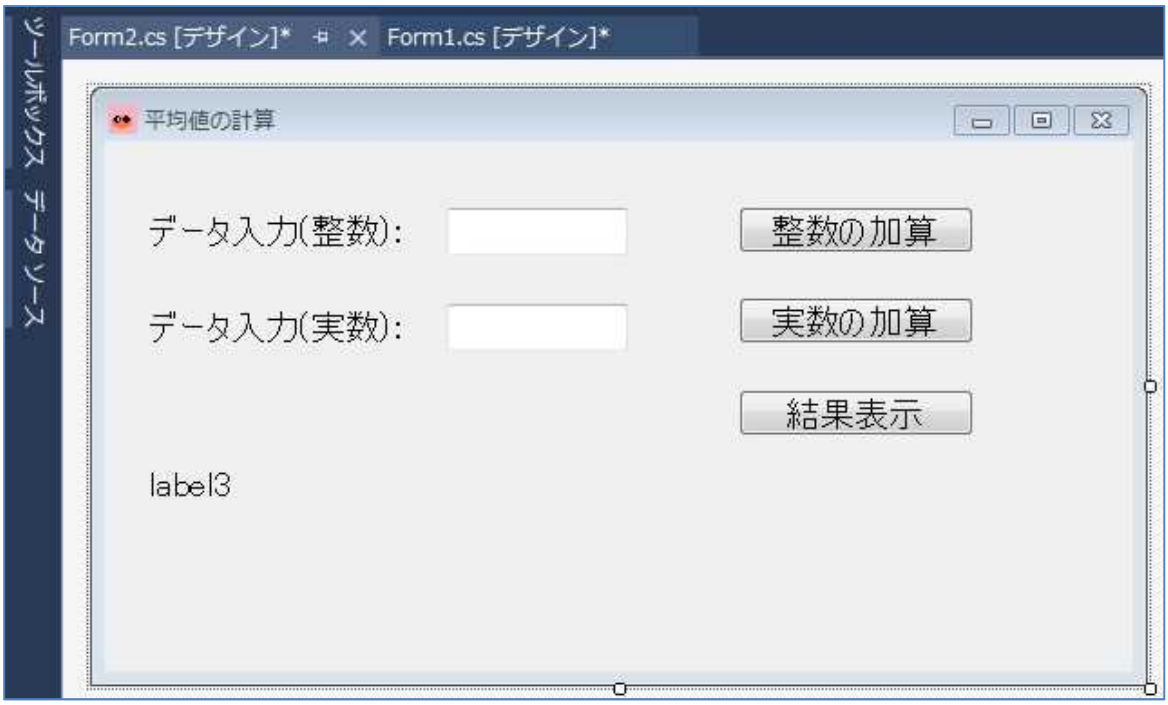
『Visual C#アイテム』及び『Windows フォーム』が選択されていることを確認して，『追加』を押す。



追加された Form2 上にラベルを 3 つ，テキストボックスを 2 つ，ボタンを 3 つ貼り付ける。それぞれのプロパティは次の様に設定する。

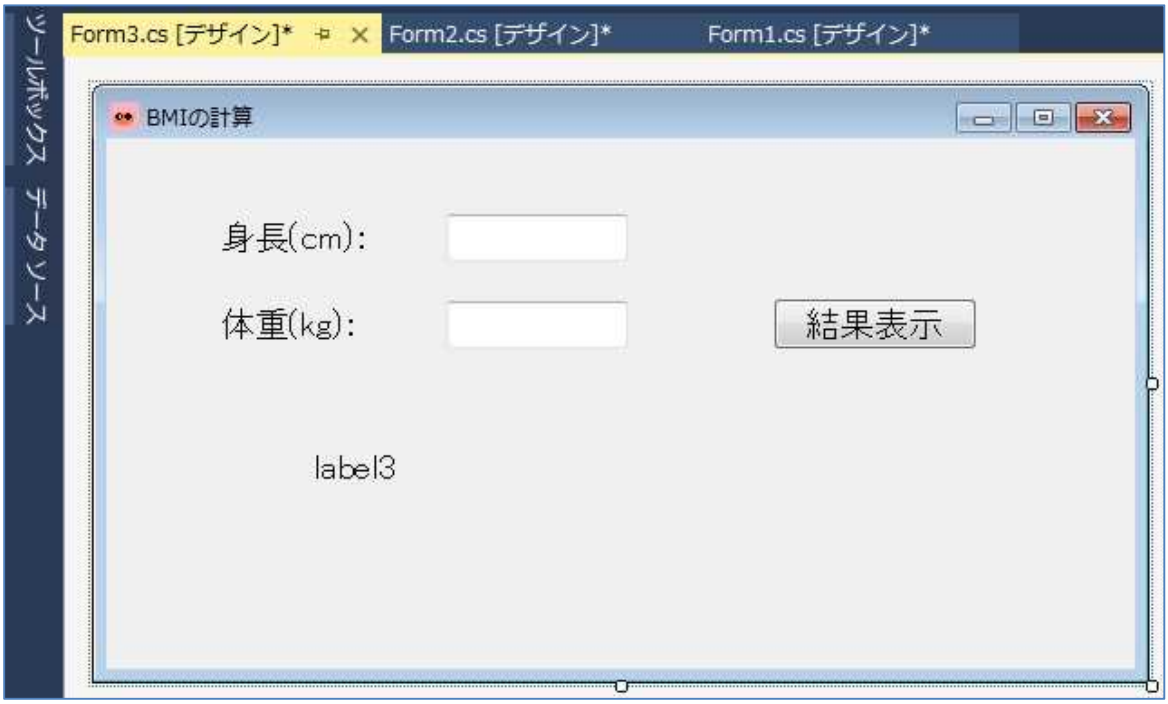
【Form2】Text　「**平均値の計算**」
　　Icon　自作のアイコン
【label1】Text　「**データ入力（整数）：**」
【label2】Text　「**データ入力(実数)：**」
【button1】Text　「**整数の加算**」
【button2】Text　「**実数の加算**」

【button3】Text 「**結果表示**」



更に Windows フォームを追加し，Form3 上にラベルを 3 つ，テキストボックスを 2 つ，ボタンを 1 つ貼り付ける。それぞれのプロパティは次の様に設定する。

【Form3】Text 「**BMI の計算**」
　　Icon　自作のアイコン
【label1】Text 「**身長(cm)：**」
【label2】Text 「**体重(kg)：**」
【button1】Text 「**結果表示**」



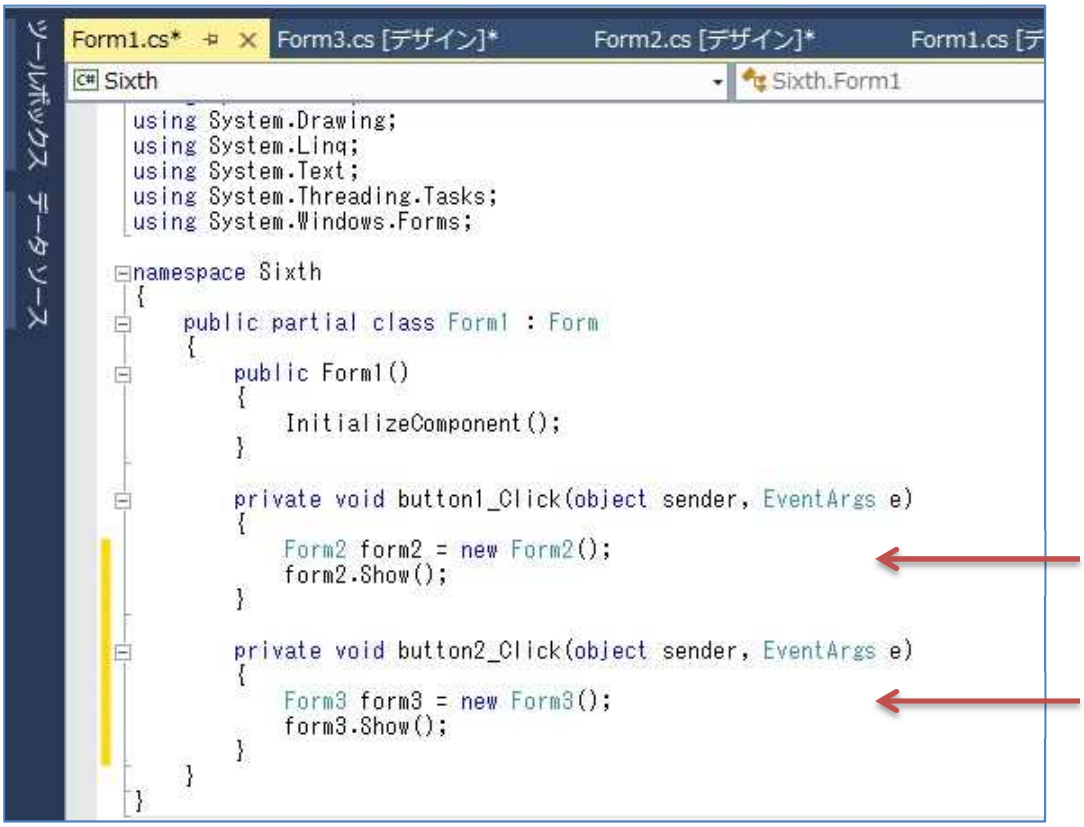
3) コーディング

Form1 のフォームデザイナー上で『button1』をダブルクリックして，Form1.cs のプログラムのソースコードを表示する。更に，フォームデザイナー上で『button2』をダブルクリックして，コードに 2 つのイベントハンドラを作成する。button1_Click メソッド及び button2_Click メソッドのブロック内にそれぞれのボタンがクリックされた際の処理（赤枠の部分）を記述していく。

```
private void button1_Click(object sender, EventArgs e)
{
    Form2 form2 = new Form2();
    form2.Show();
}

private void button2_Click(object sender, EventArgs e)
{
    Form3 form3 = new Form3();
    form3.Show();
}
```

button1_Click メソッドでは，Form2 のインスタンス（実体）を作成してそれを扱う変数を form2 とし，Show メソッドを適用して表示する処理を記述している。button2_Click メソッドでは同様にして，Form3 のインスタンスを表示する処理を記述している。



Form2 のフォームデザイナー上でコントロールが貼られていない箇所をダブルクリックして Form2.cs のプログラムのソースコードを表示する。Form2_Load メソッドのブロック内に Form2

が読み込まれた際の処理として，『label3』の『Text』プロパティに空の文字列（""）を設定する処理（赤枠の部分）を記述する。

```
private void Form2_Load(object sender, EventArgs e)
{
    label3.Text = "";
}
```

また，クラス全体に適用可能な変数の宣言をクラスの冒頭に記述する。count はデータ数を数えるためのカウンタ，sum はデータを足し込んで合計を求めるための変数である。

```
int count = 0;
double sum = 0.0;
string s;
```

更に，フォームデザイナー上で 3 つのボタンをそれぞれダブルクリックして，コードに 3 つのイベントハンドラを作成する。

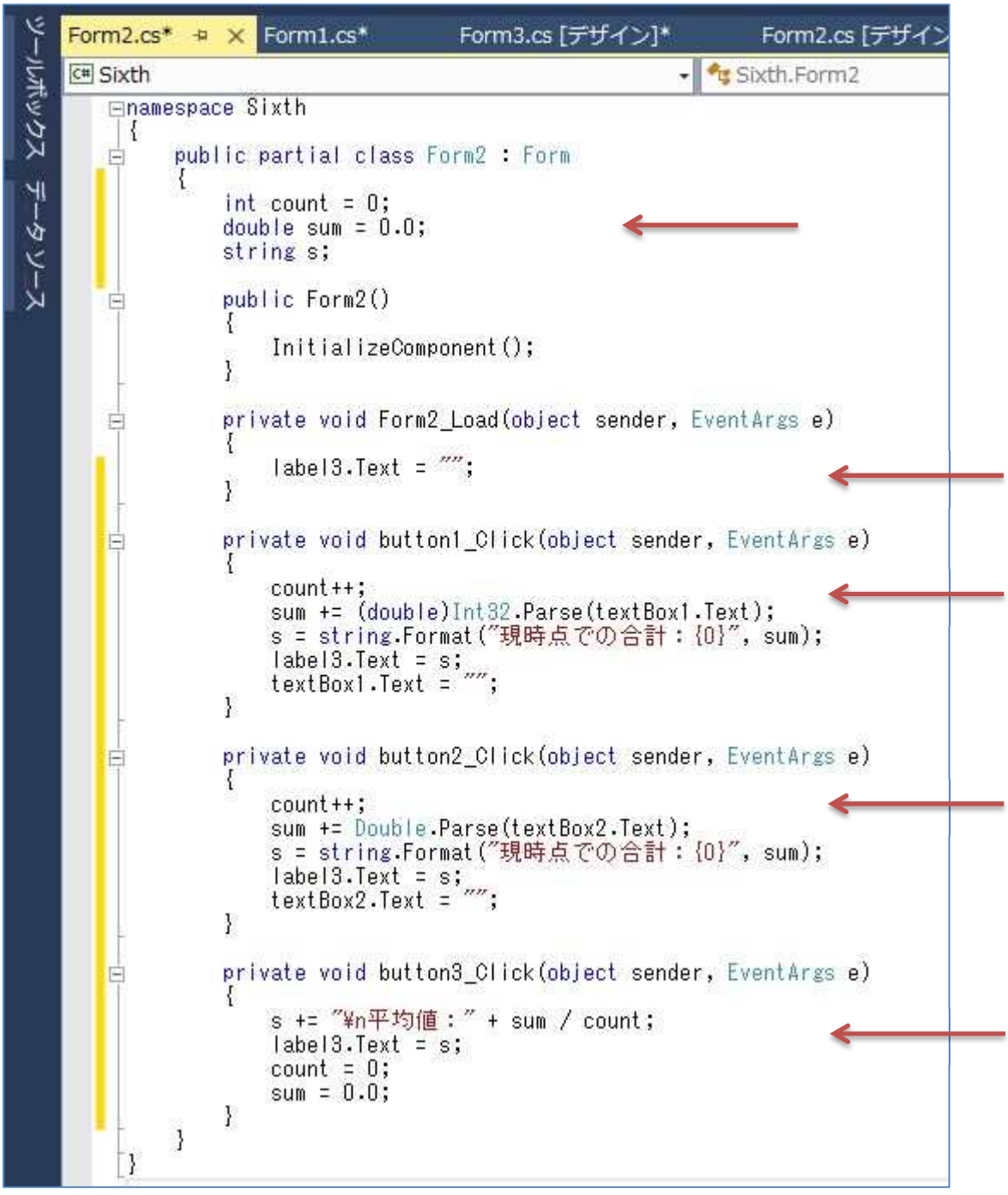
```
private void button1_Click(object sender, EventArgs e)
{
    count++;
    sum += (double)Int32.Parse(textBox1.Text);
    s = string.Format("現時点での合計：{0}", sum);
    label3.Text = s;
    textBox1.Text = "";
}

private void button2_Click(object sender, EventArgs e)
{
    count++;
    sum += Double.Parse(textBox2.Text);
    s = string.Format("現時点での合計：{0}", sum);
    label3.Text = s;
    textBox2.Text = "";
}

private void button3_Click(object sender, EventArgs e)
{
    s += "¥n 平均値：" + sum / count;
    label3.Text = s;
    count = 0;
    sum = 0.0;
}
```

button1_Click メソッド及び button2_Click メソッドでは，テキストボックスに入力された文字列を数値として読み取り，合計を求めて label3 に表示する。button1_Click メソッド中の(double)は int 型の値をこの箇所だけ double 型の値に変換する**キャスト**である。count++は，ボタンが押される度に値を 1 ずつ増やしてデータ数を数えている。

button3_Click メソッドで用いられている「+=」は p.1 に挙げた演算子よりも優先順位の低い**代入演算子**で，「s += ・・・」は「s = s + ・・・」と同じ作用がある。「+ sum / count」は除算の演算子の方が文字列の連結の演算子よりも優先順位が高いことを利用している。加算の結果を表示したい場合には，第 2 回の教材の様に「+ (a + b)」とする必要がある。



Form3 のフォームデザイナー上でコントロールが貼られていない箇所をダブルクリックして Form3.cs のプログラムのソースコードを表示する。Form3_Load メソッドのブロック内に Form3 が読み込まれた際の処理として，『label3』の『Text』プロパティに空の文字列（""）を設定する

処理（赤枠の部分）を記述する。

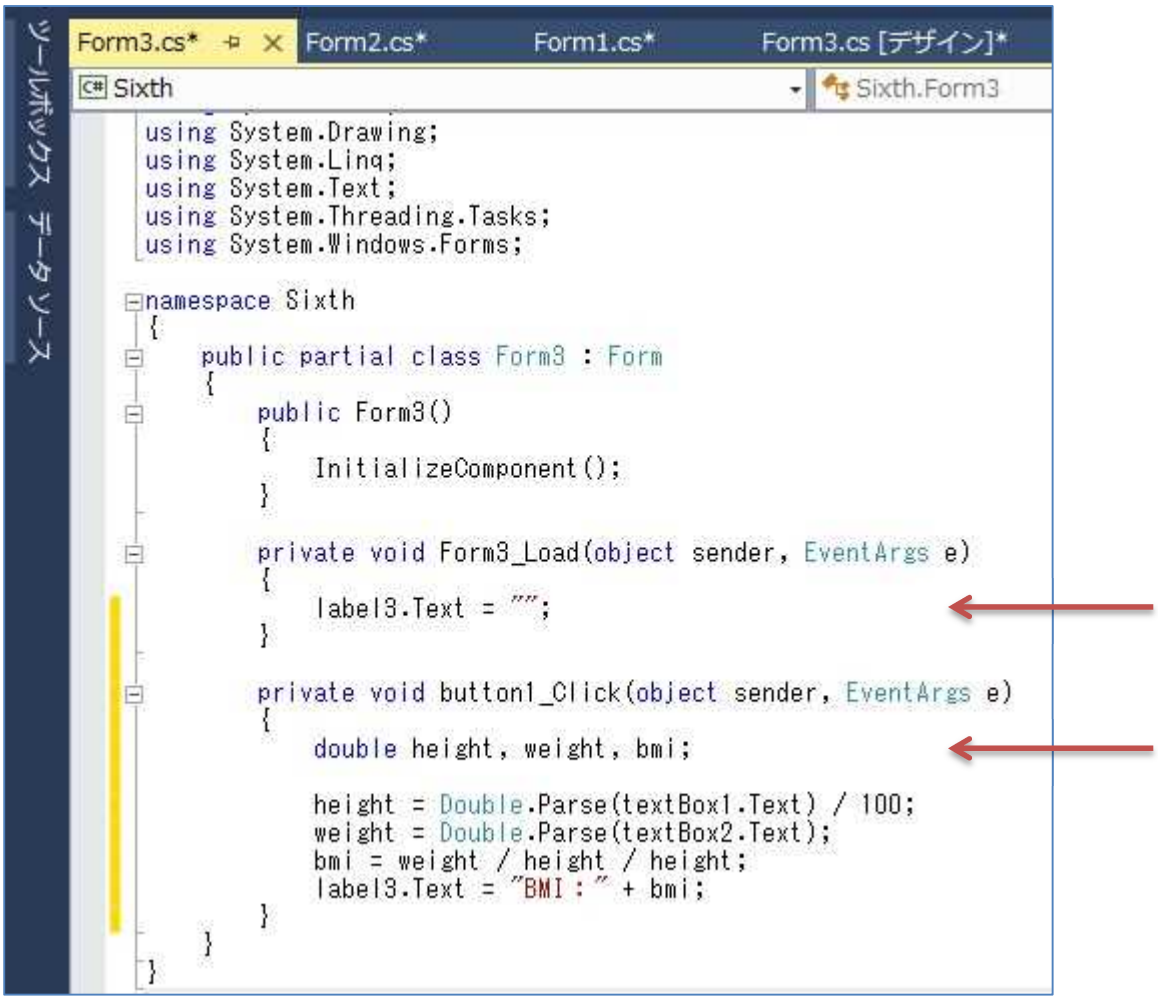
```
private void Form3_Load(object sender, EventArgs e)
{
    label3.Text = "";
}
```

更に，フォームデザイナー上でボタンをダブルクリックして，コードにイベントハンドラを作成する。

```
private void button1_Click(object sender, EventArgs e)
{
    double height, weight, bmi;

    height = Double.Parse(textBox1.Text) / 100;
    weight = Double.Parse(textBox2.Text);
    bmi = weight / height / height;
    label3.Text = "BMI : " + bmi;
}
```

button1_Click メソッドの中で宣言された変数はこのメソッドのブロック内だけに適用される。

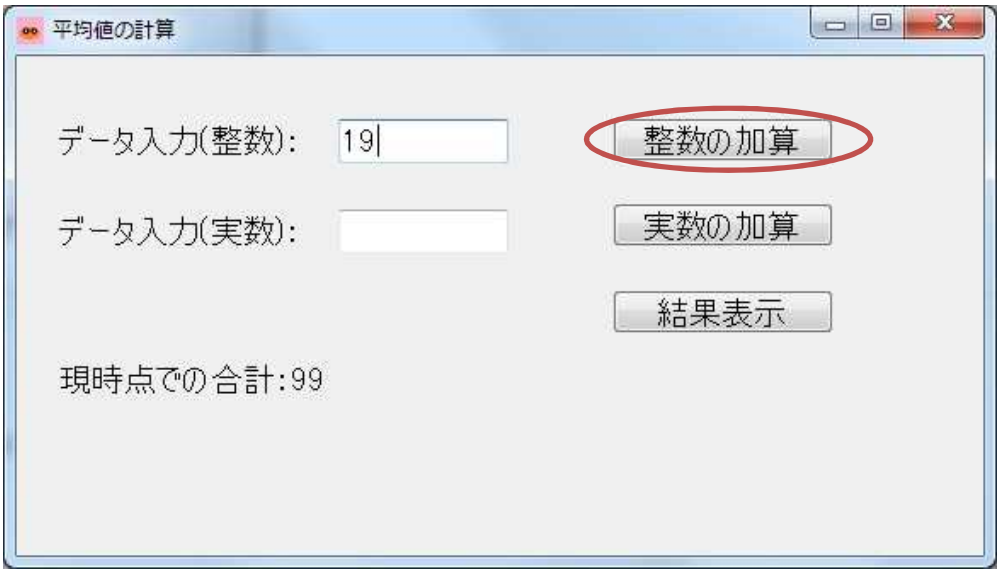


4) プログラムの実行・最終確認

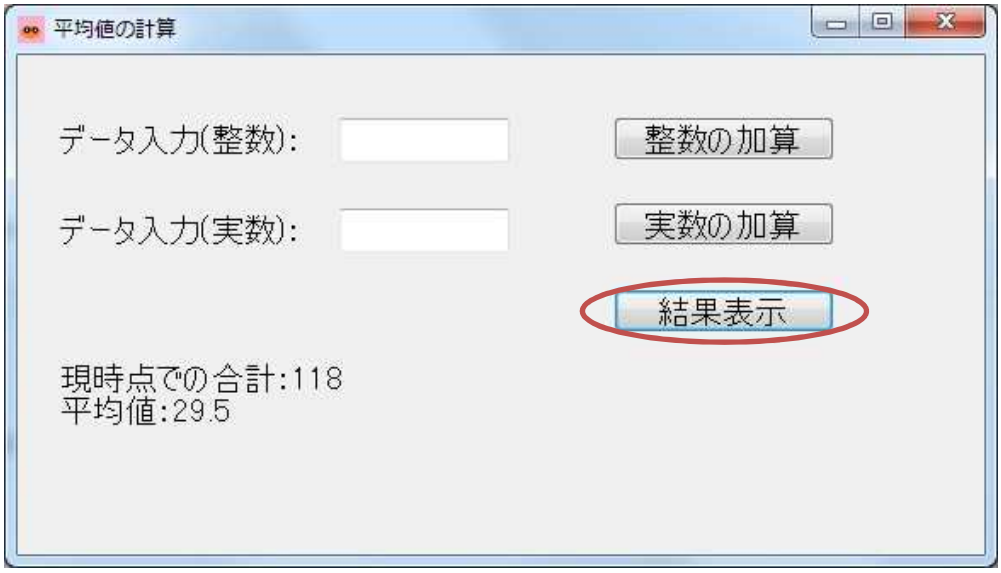
『すべてを保存』ボタンを押してから，『開始』ボタンを押して，プログラムを実行する。
エラーが出ている場合には，修正してから保存，開始と進む。

先ず，「平均値の計算」と表示されている button1 をクリックして，Form2 を表示し，その動作を確かめる。

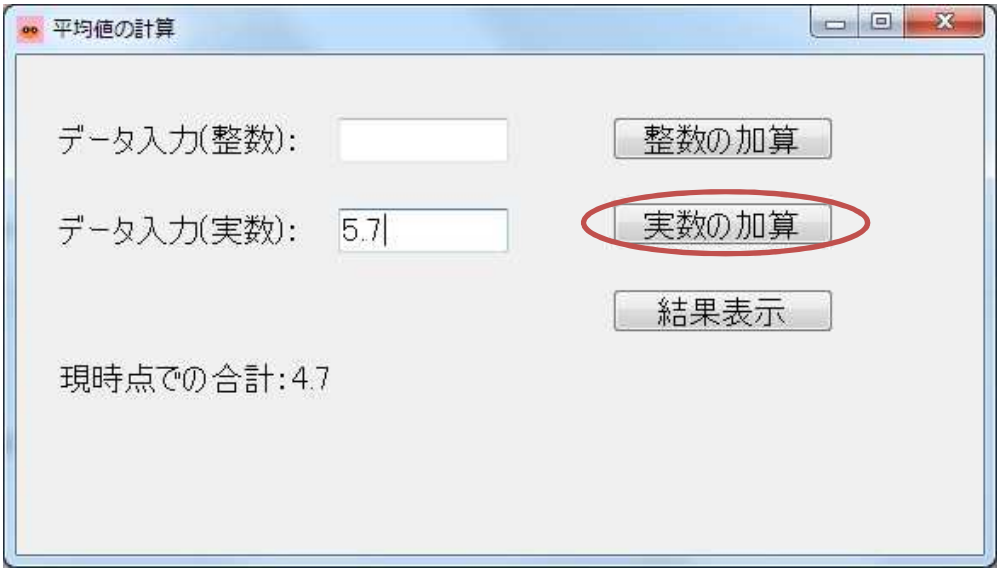
「データ入力(整数)：」と表示されたラベルの右のテキストボックスに整数値を入力して，『**整数の加算**』ボタンをクリックして，ラベルに「現時点での合計」が表示されることを確認する。

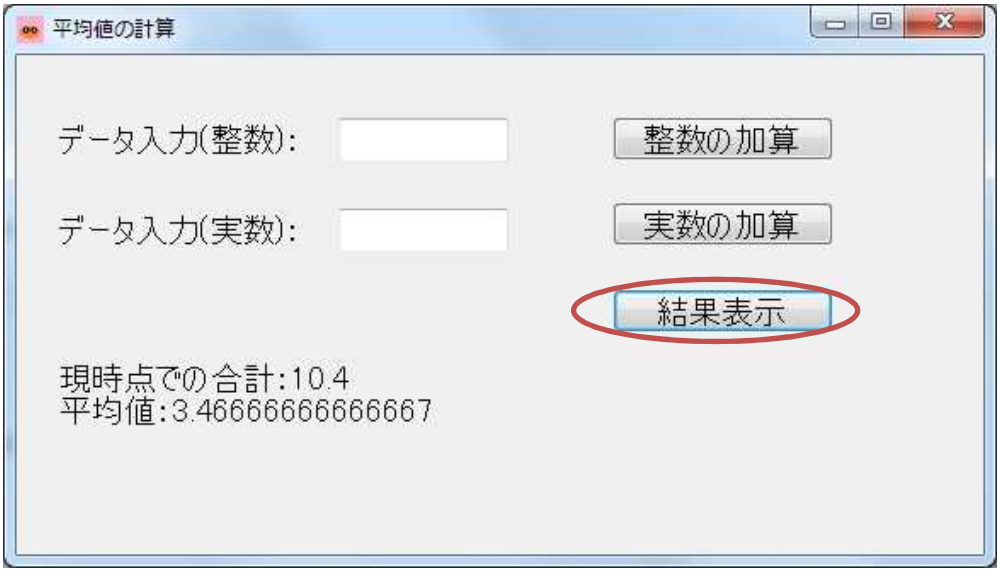


整数値の入力と『**整数の加算**』ボタンのクリックとを 10 回程度繰り返してから，『**結果表示**』ボタンをクリックして，ラベルに計算結果が表示されることを確認する。



同様に，実数についてもデータを加算していき，平均値の結果表示を確かめる。

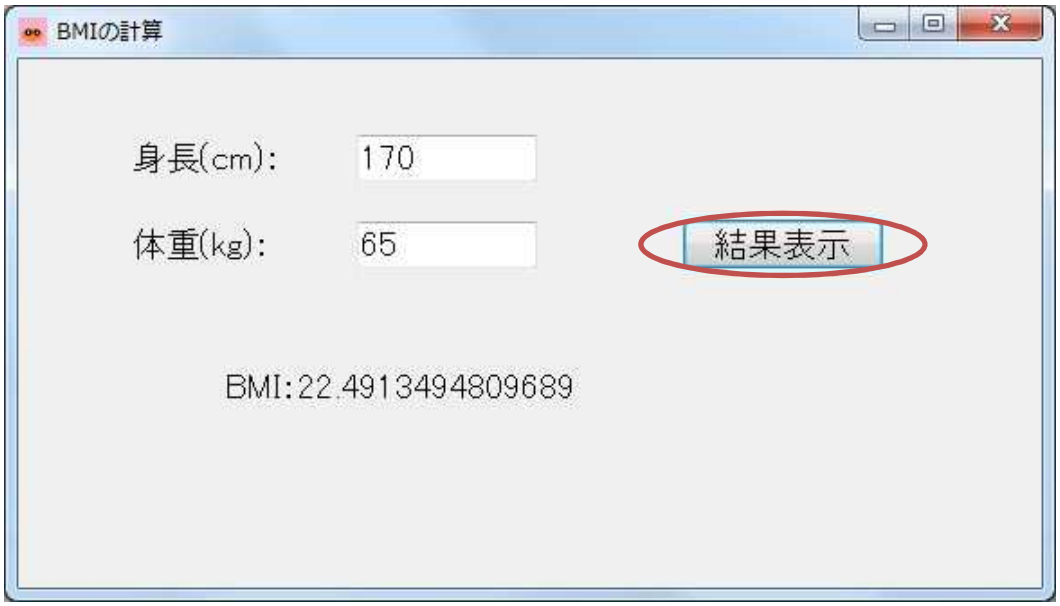




確認を終えたら，「平均値の計算」と表示された Form2 の窓のみを閉じる。

次に「ポータル」と表示された Form1 で「BMI の計算」と表示されている button1 をクリックして，Form2 を表示し，その動作を確かめる。

「身長(cm)：」及び「体重(kg)：」と表示されたそれぞれのラベルの右側のテキストボックスに値を入力して，『**結果表示**』ボタンをクリックして，ラベルに計算結果が表示されることを確認する。



確認を終えたら，プログラムを終了する。

【ファイルが保存されている場所】 H:¥Documents¥Visual Studio 2013¥Projects¥Sixth¥Sixth

提出物：

1) フォームのデザインファイル Form1.Designer.cs をメールに添付して提出する。
2) フォームを含むソースファイル Form1.cs をメールに添付して提出する。
3) フォームのデザインファイル Form2.Designer.cs をメールに添付して提出する。
4) フォームを含むソースファイル Form2.cs をメールに添付して提出する。
5) フォームのデザインファイル Form3.Designer.cs をメールに添付して提出する。
6) フォームを含むソースファイル Form3.cs をメールに添付して提出する。
7) 質問を記述したファイル Questions_6th.txt に解答を書き込んで保存し，メールに添付して提出する。