2019 年 1 月 24 日（木）実施

メニュー

メニューバーとコンテクストメニュー

Visual C#では，メニューはコントロールの一つとして扱われ，フォームアプリケーションの上部に配置される**メニューバー**と，コントロール上でマウスを右クリックすると表示される**コンテクストメニュー**とに対応している。これ等は選択するとメニューアイテムのリストが表示される**プルダウンメニュー**と呼ばれる形式に従う。

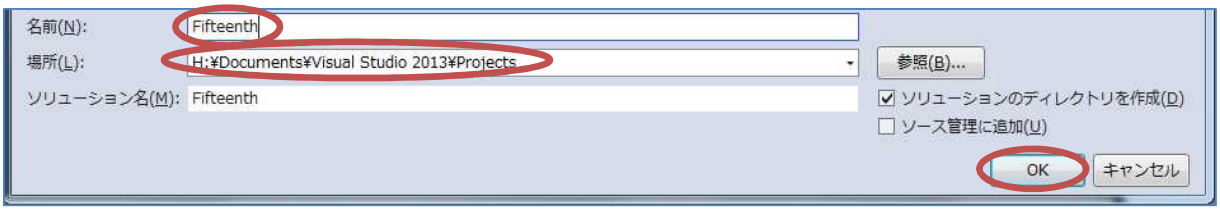
本日の課題

メニューバー及びコンテクストメニューの扱いについて，実例を通じて学ぶ。

手順

1）プロジェクトの作成

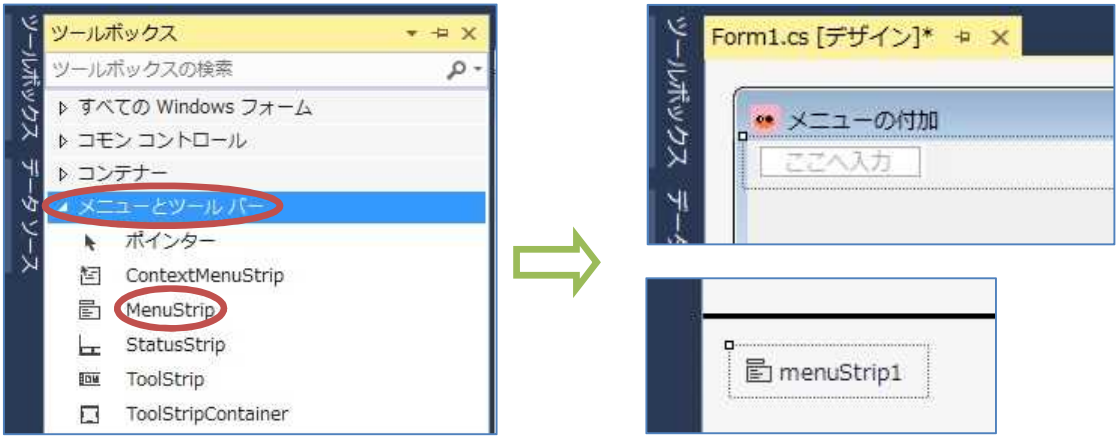
Visual Studio 2013 を起動したら，［ファイル］→［新規作成］→［プロジェクト］と辿って，プロジェクトを作成する。『新しいプロジェクト』ダイアログボックスでは，プログラミング言語を『Visual C#』，プロジェクトテンプレートとしては，『Windows フォームアプリケーション』を選択し，『名前』を「Fifteenth」に書き換え，『場所』が「H:¥Documents¥Visual Studio 2013¥Projects」となっていることを確認してから『OK』を押す（詳細は第 1 回の教材を参照）。

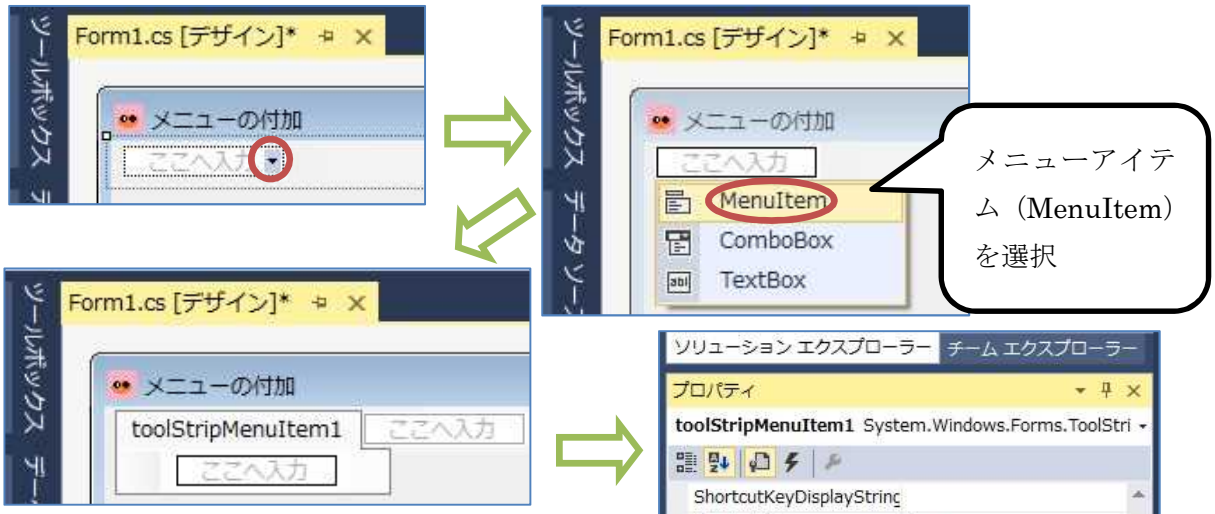


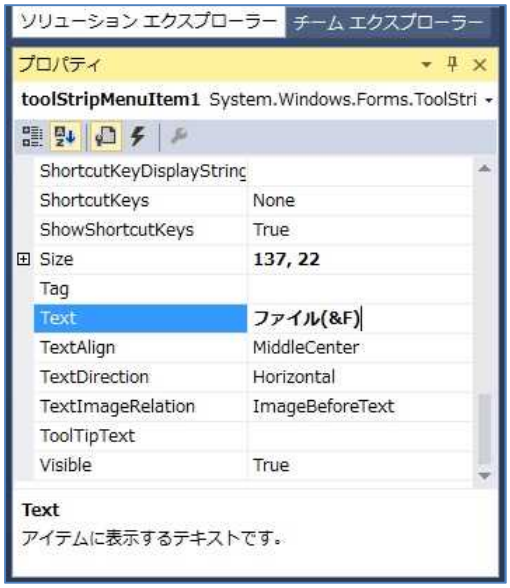
2）コントロールの配置及びフォームの作成

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，メニューアイテムに関するもの以外は，全て 14 ポイントに変更するものとする。

ツールバーの『メニューとツールバー』カテゴリからメニューストリップ（MenuStrip）を選択して，Form1 上に貼り付ける。Form1 の Text プロパティは「メニューの付加」とする。



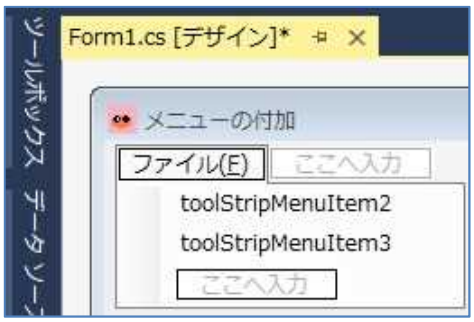




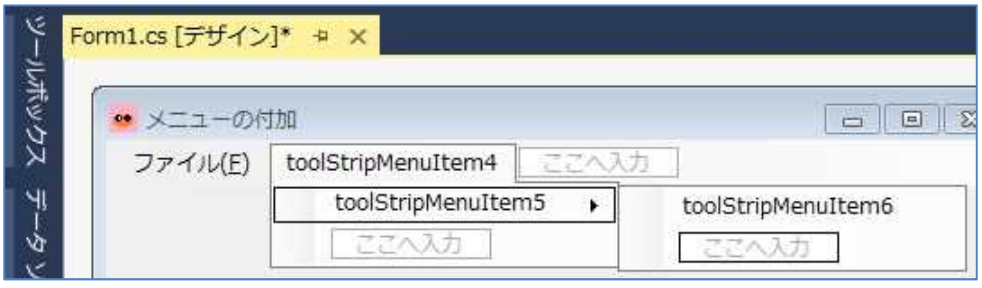
toolStripMenuItem1 の Text プロパティに「ファイル(&F)」を設定する。ここで,「(&F)」の箇所は直接入力で入力する。この設定を行うと，アプリケーションの実行時には,『Alt』のキーを押しながら『F』のキーを打つことによりこのアイテムにアクセスすることが出来る様になる。この仕組みを**アクセスキー**と呼ぶ。

「ファイル」のサブメニューとして，2 つのメニューアイテムを追加する。それぞれのプロパティは次の様に設定する。

【toolStripMenuItem2】
　Text　「開く(&O)」
【toolStripMenuItem3】
　Text　「終了(&X)」



「ファイル」の右側にメニューアイテムを 1 つ追加し，そのサブメニュー及びそこから派生したメニューとしてそれぞれ 1 つずつのメニューアイテムを追加する。



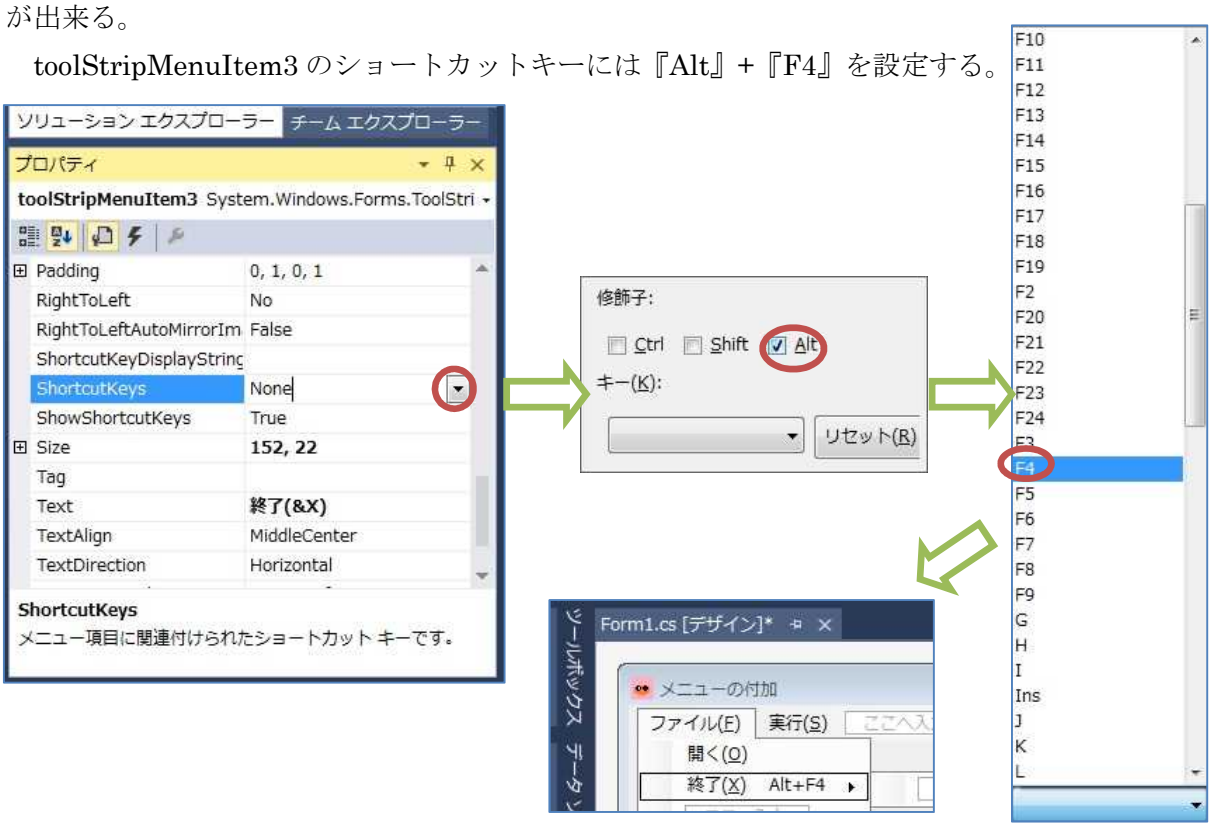
それぞれのプロパティは次の様に設定する。

【toolStripMenuItem4】　Text　「実行(&S)」

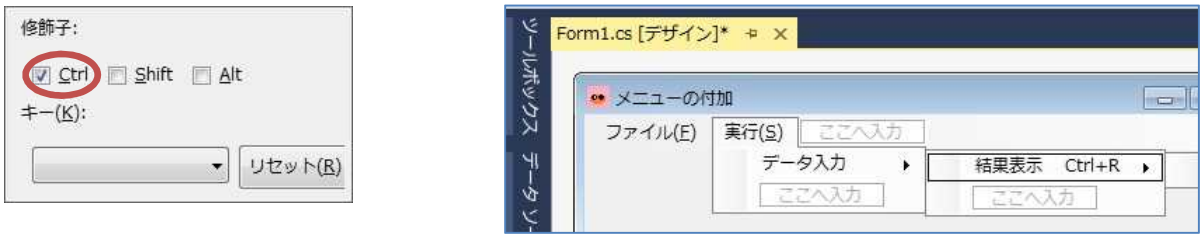
【toolStripMenuItem5】　Text　「データ入力」
【toolStripMenuItem6】　Text　「結果表示」

次に，メニューアイテムに**ショートカットキー**を設定していく。ショートカットキーを用いれば，アプリケーション実行時に階層的なメニューを辿らなくてもその内容を簡単に実行することが出来る。

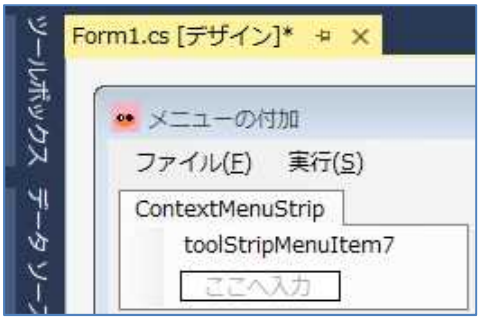
toolStripMenuItem3 のショートカットキーには『Alt』+『F4』を設定する。



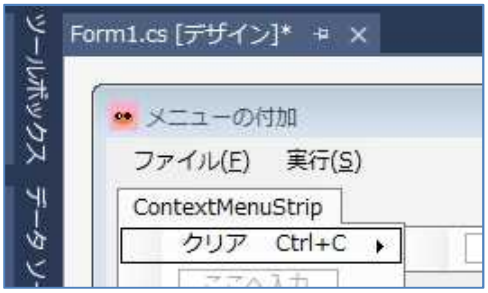
toolStripMenuItem6 のショートカットキーには，同様にして，『Ctrl』+『R』を設定する。



ツールバーの『メニューとツールバー』カテゴリからコンテクストメニューストリップ（ContextMenuStrip）を選択して，Form1 上に貼り付ける。

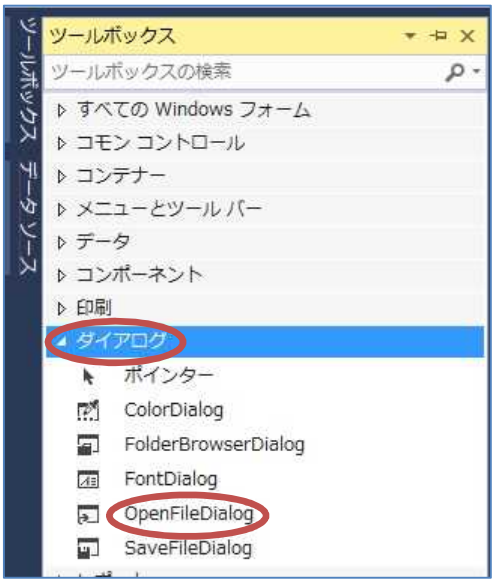


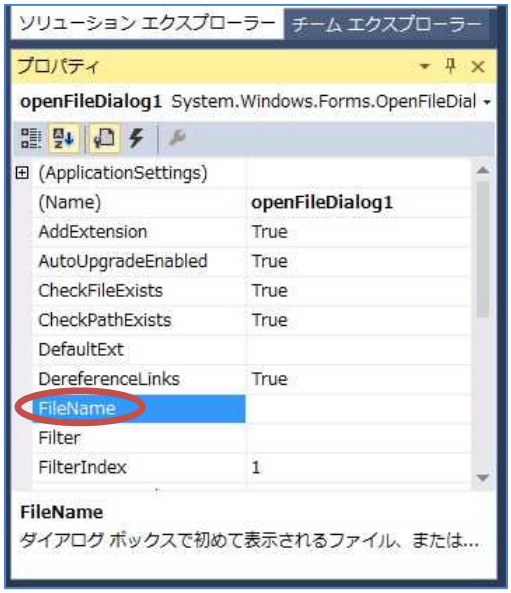
toolStripMenuItem7 の Text プロパティに「クリア」を設定し，ショートカットキーには，『Ctrl』+『C』を設定する。



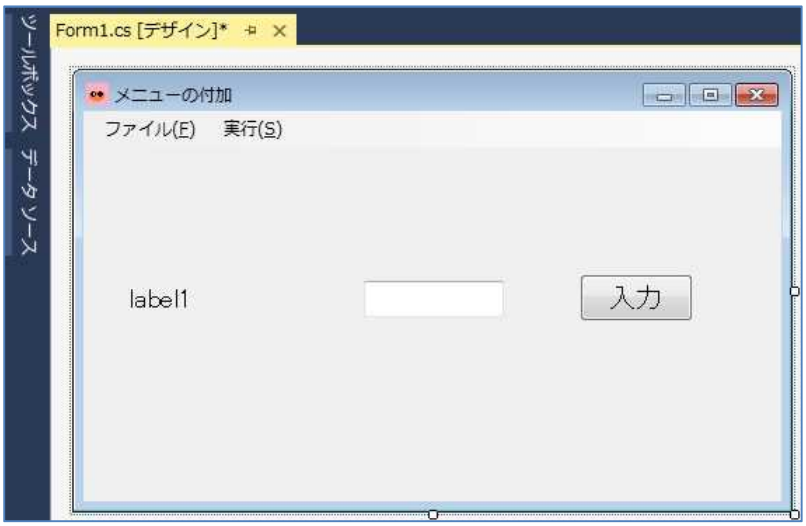
ツールボックスの『ダイアログ』カテゴリからオープンファイルダイアログ (OpenFileDialog) を選択して，Form1 上に貼り付ける。なお，Form1 のデザイン上には何も表示されないので，区切り線より下に『openFileDialog1』が表示されていることを確認する。

openFileDialog1 の FileName プロパティに設定されている文字列は削除して，空にしておく。

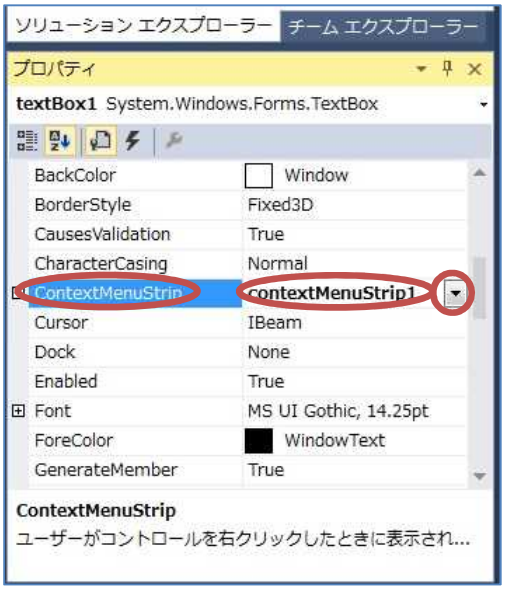




Form1 上に，ラベルを 1 つ，テキストボックスを 1 つ，ボタンを 1 つ貼り付ける。button1 の Text プロパティに「**入力**」を設定する。



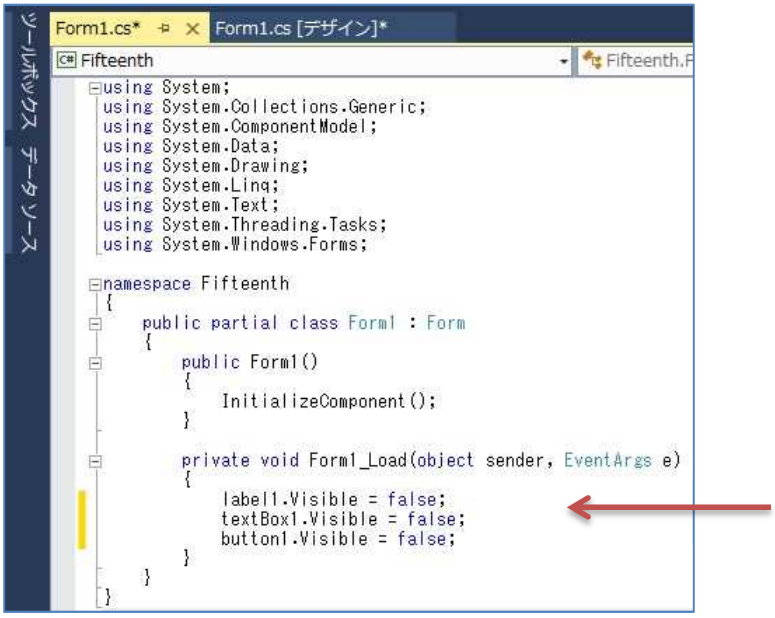
textBox1 の ContextMenuStrip プロパティで，下向き三角ボタン（▼）を押し出て来る候補から「contextMenuStrip1」を選択して設定する。



3) コーディング

Form1 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 Form1.cs のプログラムのソースコードを表示する。Form1_Load メソッドのブロック内に Form1 が読み込まれた際の処理として，『label1』，『textBox1』及び『button1』の『Visible』プロパティに「false」を設定して非表示にする処理（赤枠の部分）を記述する。

```
        private void Form1_Load(object sender, EventArgs e)
        {
            label1.Visible = false;
            textBox1.Visible = false;
            button1.Visible = false;
        }
```



Form1 のフォームデザイナー上で『toolStripMenuItem2』，『toolStripMenuItem3』，『toolStripMenuItem5』及び『toolStripMenuItem7』をダブルクリックして，それぞれのメニューアイテム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 toolStripMenuItem2_Click(object sender, EventArgs e)
{
    openFileDialog1.ShowDialog();
}
```

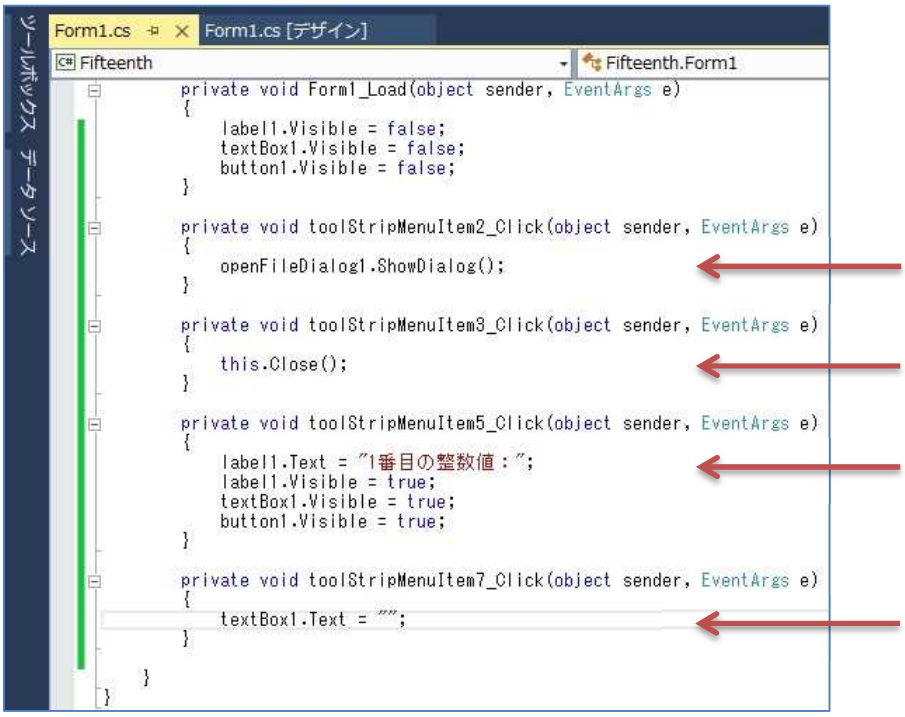
ShowDialog()で，ファイルを開く為のダイアログを表示する。

```
private void toolStripMenuItem3_Click(object sender, EventArgs e)
{
    this.Close();
}
```

this.Close()で，このフォームを閉じ，フォームアプリケーションを終了する。

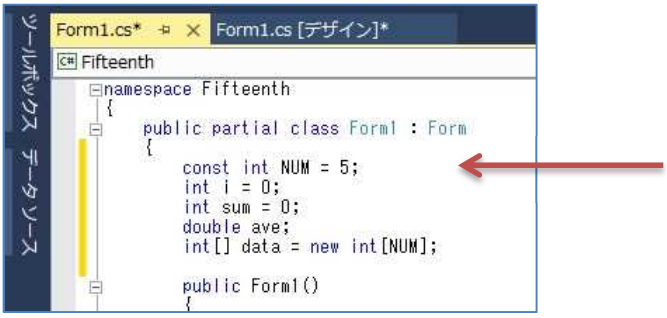
```
private void toolStripMenuItem5_Click(object sender, EventArgs e)
{
    label1.Text = "1 番目の整数値：";
    label1.Visible = true;
    textBox1.Visible = true;
    button1.Visible = true;
}
```

```
private void toolStripMenuItem7_Click(object sender, EventArgs e)
{
    textBox1.Text = "";
}
```



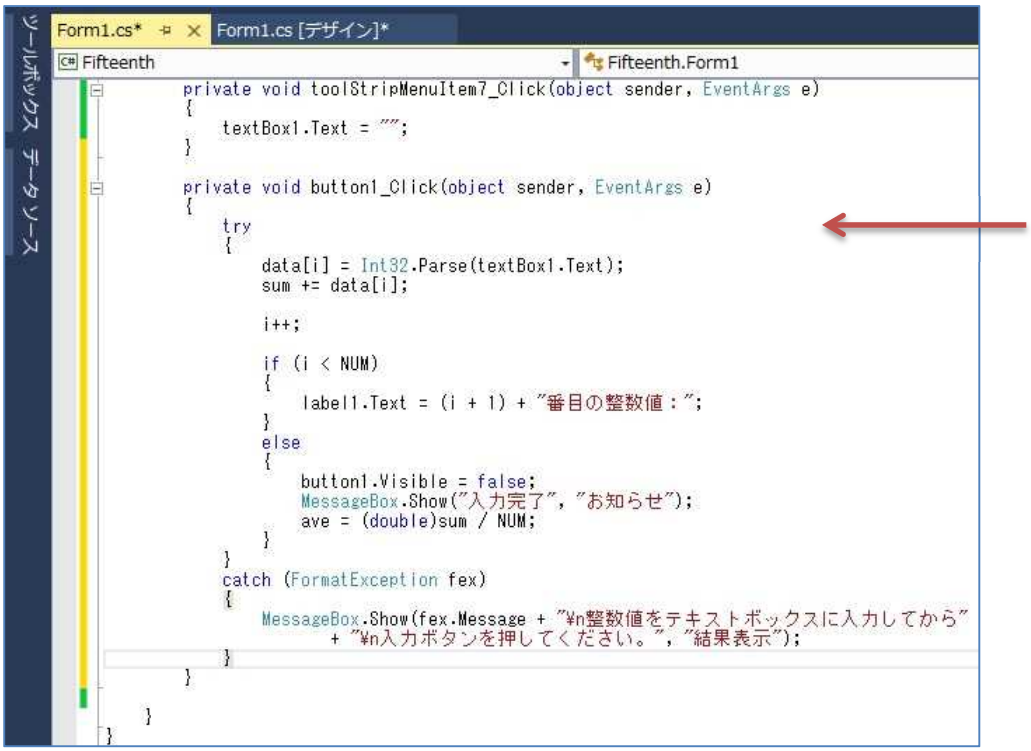
Form1 のフォームデザイナー上で『button1』をダブルクリックして，Form1.cs のプログラムのソースコードを表示する。先ず，クラス全体に適用可能な定数，変数及び配列の宣言をクラスの冒頭に記述する。

```
const int NUM = 5;
int i = 0;
int sum = 0;
double ave;
int[] data = new int[NUM];
```



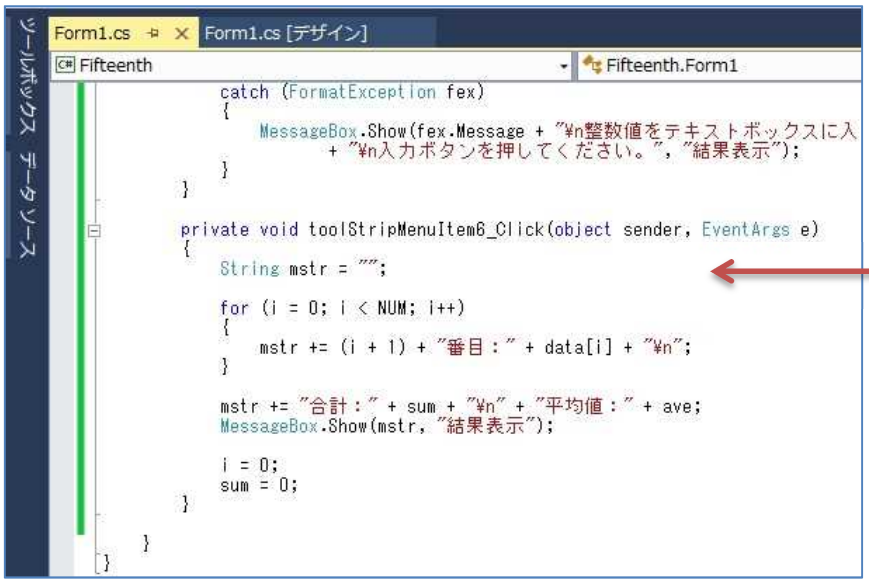
次に，button1_Click メソッドのブロック内にボタンがクリックされた際の処理（赤枠の部分）を記述していく。

```
private void button1_Click(object sender, EventArgs e)
{
    try
    {
        data[i] = Int32.Parse(textBox1.Text);
        sum += data[i];

        i++;

        if (i < NUM)
        {
            label1.Text = (i + 1) + "番目の整数値：";
        }
        else
        {
            button1.Visible = false;
            MessageBox.Show("入力完了", "お知らせ");
            ave = (double)sum / NUM;
        }
    }
    catch (FormatException fex)
    {
        MessageBox.Show(fex.Message + "¥n 整数値をテキストボックスに入力してから"
                + "¥n 入力ボタンを押してください。", "結果表示");
    }
}
```



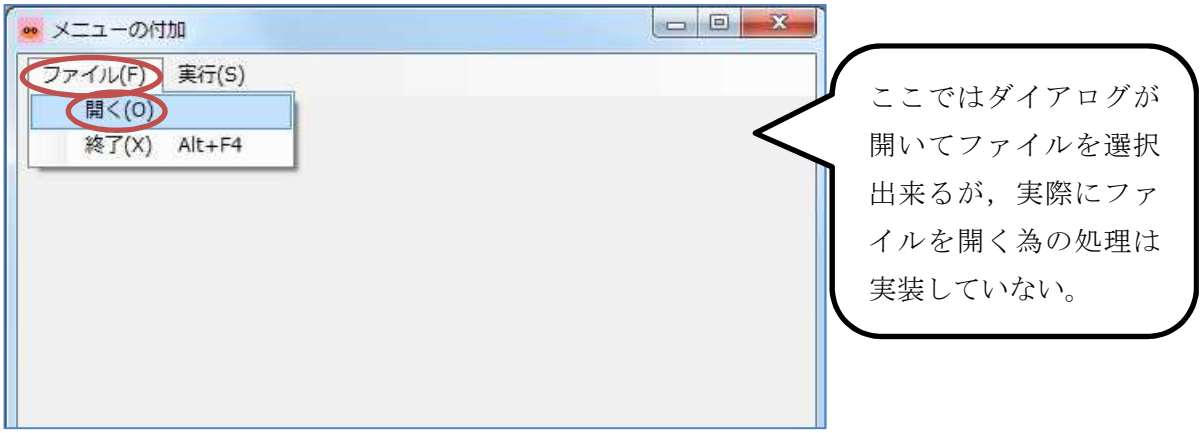
最後に，Form1 のフォームデザイナー上で『toolStripMenuItem6』をダブルクリックして，それぞれのメニューアイテムがクリックされた際の処理（赤枠の部分）を記述する。

```
private void toolStripMenuItem6_Click(object sender, EventArgs e)
{
    String mstr = "";

    for (i = 0; i < NUM; i++)
    {
        mstr += (i + 1) + "番目：" + data[i] + "¥n";
    }

    mstr += "合計：" + sum + "¥n" + "平均値：" + ave;
    MessageBox.Show(mstr, "結果表示");

    i = 0;
    sum = 0;
}
```



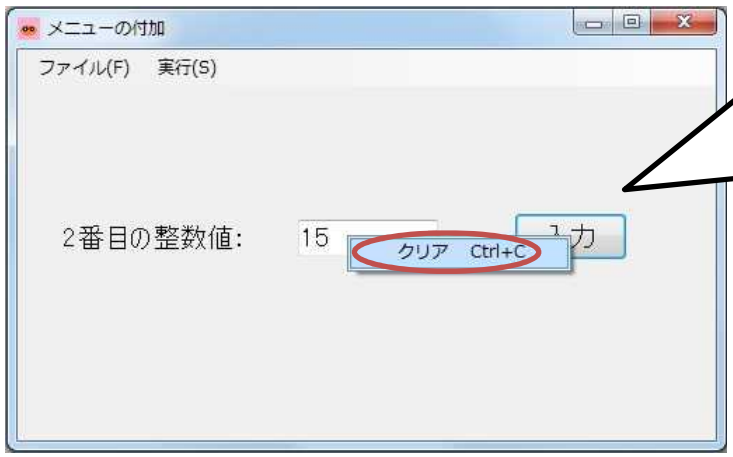
4) プログラムの実行・最終確認

『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。



メニューアイテムの「データ入力」を選択すると，ラベル，テキストボックス及びボタンが表示される。

テキストボックスに整数値を入力し，ボタンを押した後，テキストボックスを右クリックして，コンテクストメニューを表示させる。



「クリア」を選択するとテキストボックスの文字列が空になる。テキストボックスをクリックして選択し，『Ctrl』+『C』でも同じ動作をすることを確かめる。

『Ctrl』+『R』で表示

確認を終えたら，『Alt』+『F4』で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¥Fifteenth¥Fifteenth

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。