2023 年 7 月 20 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

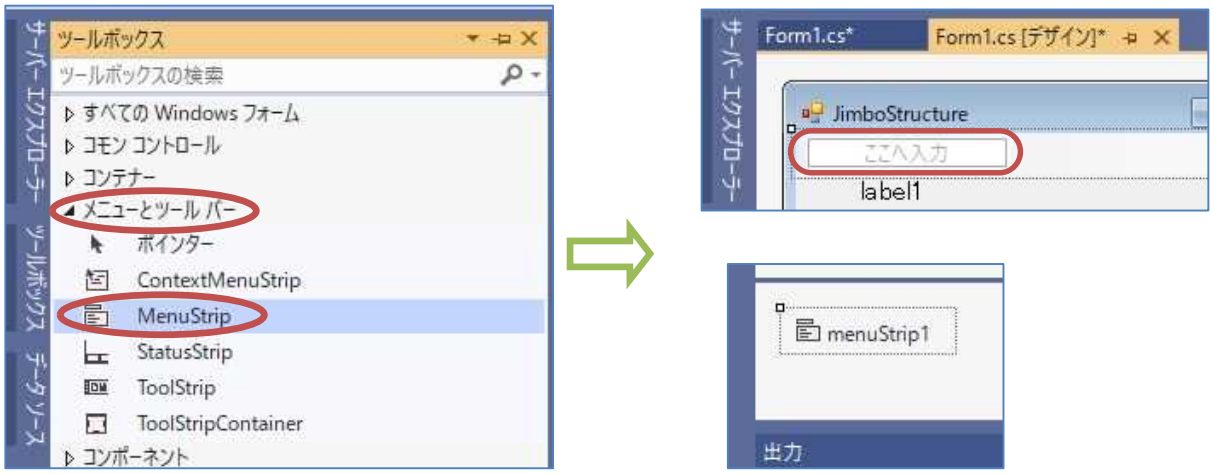
今回の授業では，アプリの作成を通じて，メニューバー及びコンテクストメニューの扱いについて学ぶ。

手順

1）コントロールの配置・プロパティの設定

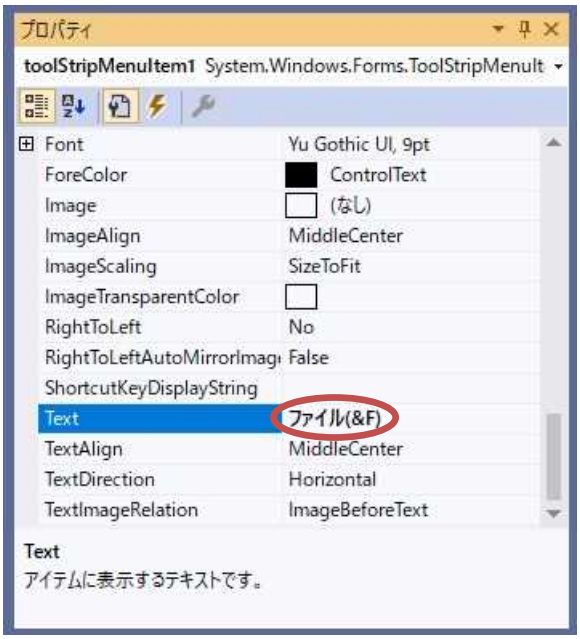
【メニューバーの配置】

前回作成したプロジェクトを開き，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。



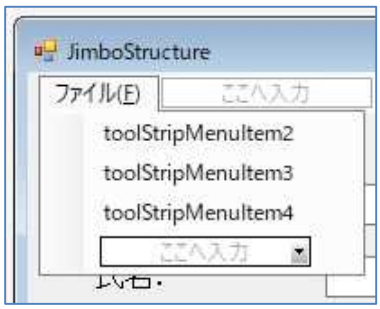
メニューストリップをクリックして表示される▼ボタンを押して，出てくるプルダウンメニューで，メニューアイテム（MenuItem）を選択する。

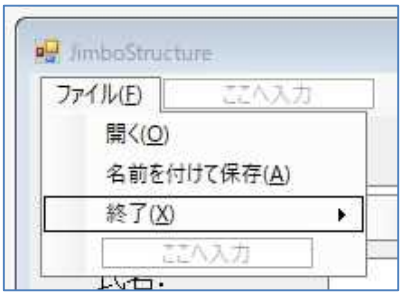
toolStripMenuItem1 の Text プロパティに「ファイル(&F)」を設定する。ここで，「(&F)」の箇所は半角文字で入力する。この設定を行うと，アプリケーションの実行時には，『Alt』のキーを押さえながら『F』のキーを打つことによりこのアイテムにアクセスすることが出来る様になる。この仕組みをアクセスキーと呼ぶ。



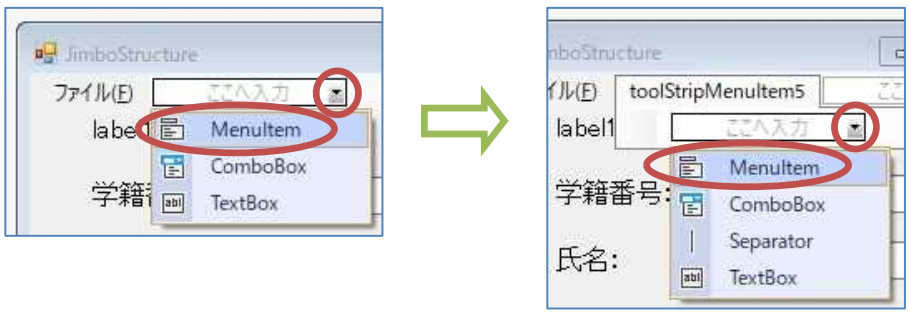
「ファイル」のサブメニューとして，3 つのメニューアイテムを追加する。それぞれのプロパティは次の様に設定する。

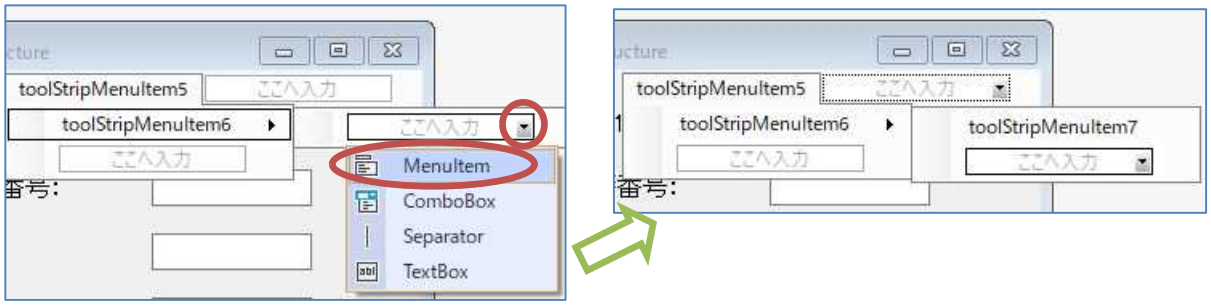
【toolStripMenuItem2】
　Text　「開く(&O)」
【toolStripMenuItem3】
　Text　「名前を付けて保存(&A)」
【toolStripMenuItem4】
　Text　「終了(&X)」





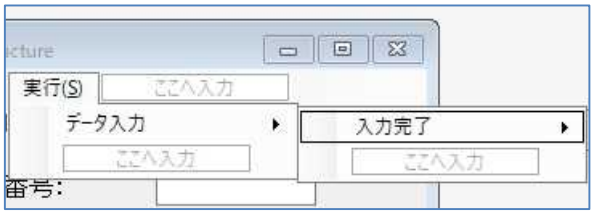
「ファイル」の右側にメニューアイテムを 1 つ追加し，そのサブメニュー及びそこから派生したメニューとしてそれぞれ 1 つずつのメニューアイテムを追加する。





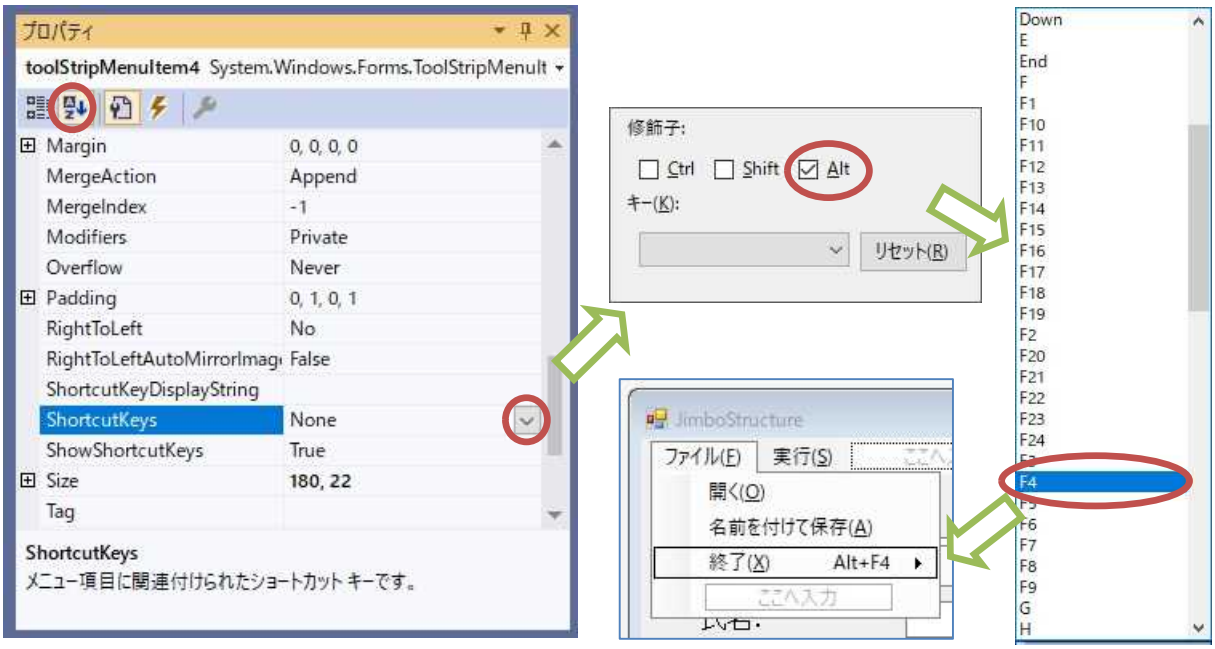
それぞれのプロパティは次の様に設定する。

【toolStripMenuItem5】
　Text　「実行(&S)」
【toolStripMenuItem6】
　Text　「データ入力」
【toolStripMenuItem7】
　Text　「入力完了」

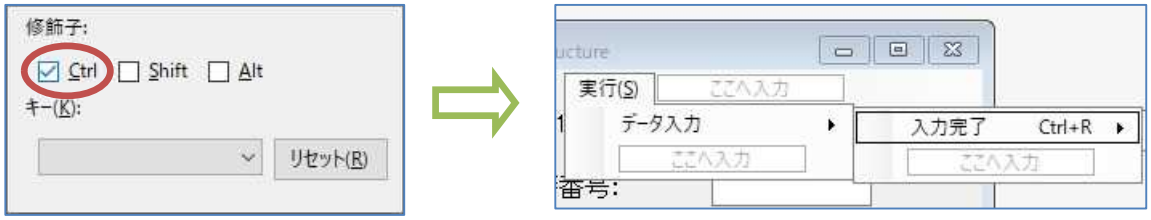


次に，メニューアイテムにショートカットキーを設定していく。ショートカットキーを用いれば，アプリケーション実行時に階層的なメニューを辿らなくてもその内容を簡単に実行することが出来る。

toolStripMenuItem4 の ShortcutKeys プロパティには『Alt』及び『F4』を設定する。

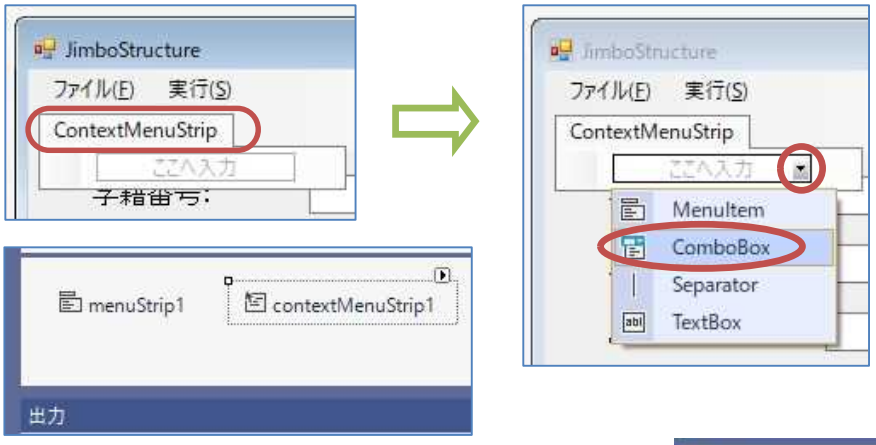


toolStripMenuItem7 のショートカットキーには，同様にして，『Ctrl』及び『R』を設定する。

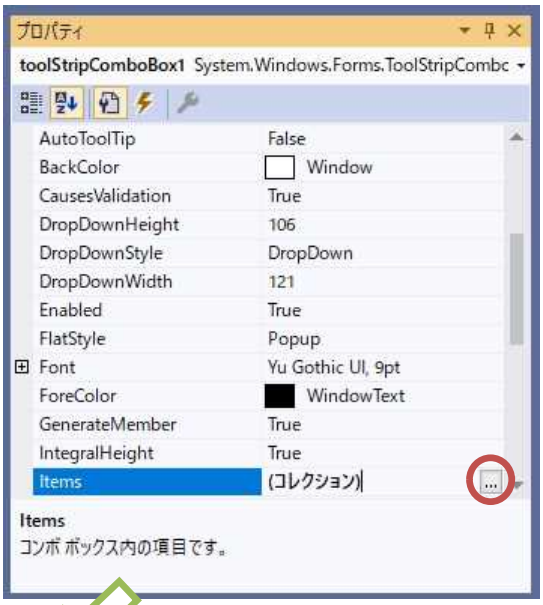


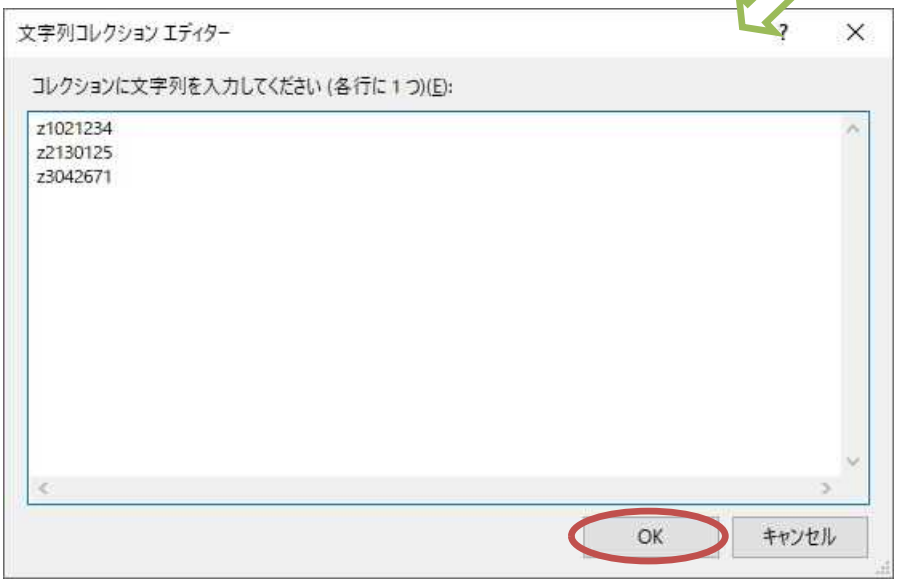
【コンテクストメニューの配置】

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，プルダウンメニューでコンボボックス（ComboBox）を選択して追加する。

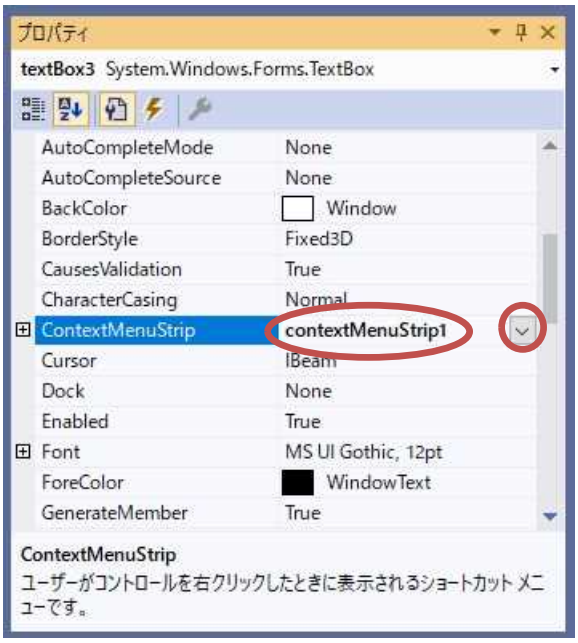


toolStripComboBox1 の Items プロパティには，『…』ボタンを押して，文字列コレクションエディターを起動し，"z1021234"，"z2130125"，"z3042671"の 3 個の履修科目コードを入力する（二重引用符は入力しない）。





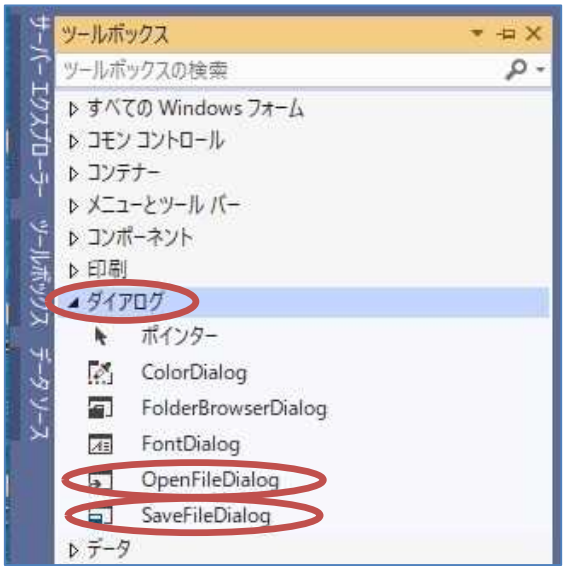
textBox3 の ContextMenuStrip プロパティで，下向き三角ボタン（▼）を押し出て来る候補から「contextMenuStrip1」を選択して設定する。

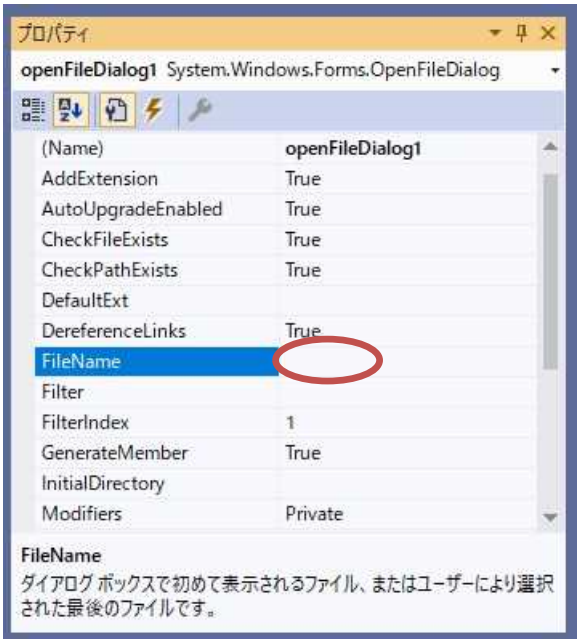


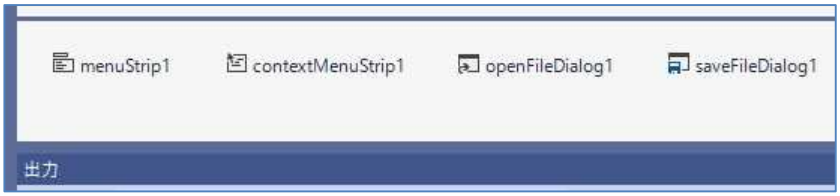
【ダイアログの配置】

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）及びセーブファイルダイアログ（SaveFileDialog）を選択して，それぞれを 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』及び『saveFileDialog1』が表示されていることを確認する。

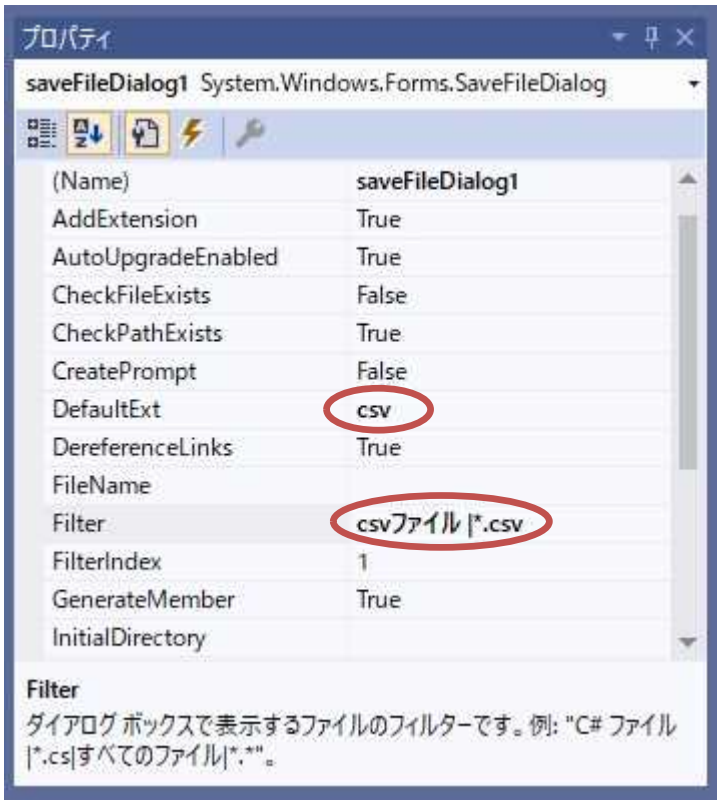
openFileDialog1 の FileName プロパティに設定されている文字列は削除して，空にしておく。







saveFileDialog1 の DefaultExt プロパティは"csv"，Filter プロパティは "csv ファイル|*.csv" に設定する。



2) コーディング

フォームデザイナー上で『toolStripMenuItem3』をダブルクリックして，メニューアイテム『名前を付けて保存』がクリックされた際のイベントハンドラの処理（赤枠の部分）を記述する。

```
        1 個の参照
115     private void toolStripMenuItem3_Click(object sender, EventArgs e)
116     {
117         int i = 1;
118         string mstr = "データ番号，学籍番号，氏名　　，履修科目コード，点数，評価";
119
120         foreach (Student list in listSt)
121         {
122             mstr += "¥n" + i + "," + list.id + "," + list.name + ", "
123                 + list.code + "," + list.point + ", " + list.eval;
124             i++;
125         }
126
127         saveFileDialog1.ShowDialog();
128         try
129         {
130             using (StreamWriter sw = new StreamWriter(saveFileDialog1.FileName))
131             {
132                 sw.WriteLine(mstr);
133             }
134         }
135         catch (Exception ex)
136         {
137             MessageBox.Show(ex.Message);
138         }
139     }
```

ここで，ShowDialog メソッドは，ファイルを開くためのダイアログを表示する。また，FileName

プロパティはダイアログで指定したファイルのパス名付きのファイル名となる。

＊ カンマとその後の空白は半角文字で入力する。なお，「氏名」の後には桁合わせのために全角空白 3 文字を入れている。

また，第 11 回の教材と同様に，using ディレクティブとして using System.IO; を加える必要がある。

```
1   using System;
2   using System.Collections.Generic;
3   using System.ComponentModel;
4   using System.Data;
5   using System.Drawing;
6   using System.IO;
7   using System.Linq;
8   using System.Text;
9   using System.Threading.Tasks;
10  using System.Windows.Forms;
```

フォームデザイナー上で『toolStripMenuItem2』をダブルクリックして，メニューアイテム『開く』がクリックされた際のイベントハンドラの処理（赤枠の部分）を記述する。ここで，EndOfStream プロパティはファイルの終わりに達すると true，それ以外では False となる。また，ReadLine メソッドはファイルから 1 行分を読み込むために用い，Split メソッドは 1 個または複数個の区切り記号に基づいて入力文字列を分割することで部分文字列の配列を作成するために用いる。

```
        1 個の参照
141     private void toolStripMenuItem2_Click(object sender, EventArgs e)
142     {
143         string mstr = "";
144
145         openFileDialog1.ShowDialog();
146         try
147         {
148             using (StreamReader sr = new StreamReader(openFileDialog1.FileName))
149             {
150                 while (sr.EndOfStream == false)
151                 {
152                     string line = sr.ReadLine();
153                     string[] fields = line.Split(',');
154
155                     for (int i = 0; i < fields.Length; i++)
156                     {
157                         mstr += fields[i] + "¥t";
158                     }
159                     mstr += "¥n";
160                 }
161             }
162         }
163         catch (Exception ex)
164         {
165             mstr = ex.Message;
166         }
167         finally
168         {
169             MessageBox.Show(mstr);
170         }
171     }
```

フォームデザイナー上で『toolStripMenuItem4』をダブルクリックして，メニューアイテム『終了』がクリックされた際のイベントハンドラの処理（赤枠の部分）を記述する。ここで，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。

```
        1 個の参照
173     private void toolStripMenuItem4_Click(object sender, EventArgs e)
174     {
175         this.Close();
176     }
```

次に，button1_Click の内容を抽出して dataInput メソッドを作成する（第 6 回教材 pp.2-4）。

```
        1 個の参照
66      private void button1_Click(object sender, EventArgs e)
67      {
68          dataInput();
69      }
70
        1 個の参照
71      private void dataInput()
72      {
73          st.id = textBox1.Text;
74          st.name = textBox2.Text;
75          st.code = textBox3.Text;
76
77          try
78          {
79              st.point = Int32.Parse(textBox4.Text);
80          }
81          catch (FormatException fex)
82          {
83              MessageBox.Show(fex.Message
84              + "¥n整数値をテキストボックスに入力してから"
85              + "¥n入力ボタンを押してください。"
86              + "¥n今回はエラーが記録されます。");
87              st.point = -1;
88          }
89          finally
90          {
91              st.eval = eva(st.point);
92              listSt.Add(st);
93
94              dnum++;
95              textBox1.Text = "";
96              textBox2.Text = "";
97              textBox3.Text = "";
98              textBox4.Text = "";
99              label1.Text = (dnum + 1) + "番目のデータ";
100         }
101     }
```

続けて，button2_Click の内容を抽出して resDisp メソッドを作成する。

（図は次のページ）

```
1 個の参照
private void button2_Click(object sender, EventArgs e)
{
    resDisp();
}

2 個の参照
private void resDisp()
{
    int i = 1;
    string mstr = "入力されたデータ";

    foreach (Student list in listSt)
    {
        mstr += "¥n" + i + ") " + "学籍番号： " + list.id
        + "¥t氏名： " + list.name + "¥t履修科目コード： "
        + list.code + "¥t点数： " + list.point
        + "¥t評価： " + list.eval;
        i++;
    }

    MessageBox.Show(mstr);
}
```

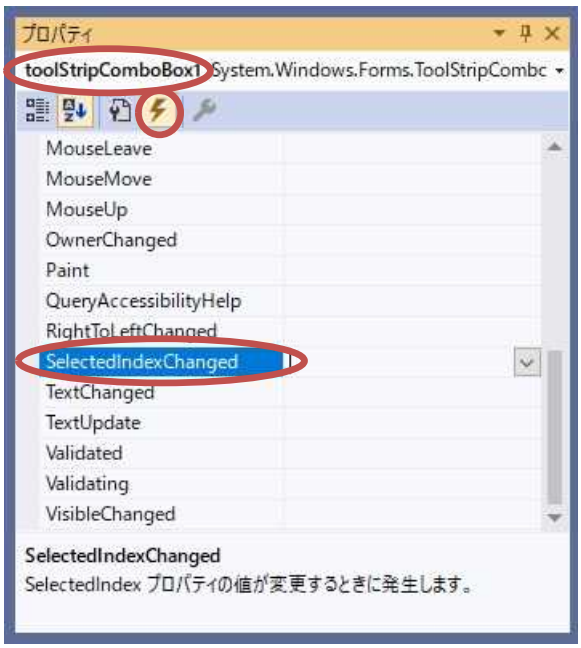
フォームデザイナー上で『toolStripMenuItem6』及び『toolStripMenuItem7』をダブルクリックして，それぞれのメニューアイテムがクリックされた際のイベントハンドラの処理（赤枠の部分）を記述する。

```
1 個の参照
private void toolStripMenuItem6_Click(object sender, EventArgs e)
{
    dataInput();
}
```

```
1 個の参照
private void toolStripMenuItem7_Click(object sender, EventArgs e)
{
    resDisp();
}
```

フォームデザイナーのプロパティウィンドウで toolStripComboBox1 を選択し，イベントボタン（稲妻のアイコン）を選択する。

SelectedIndexChanged をダブルクリックする。



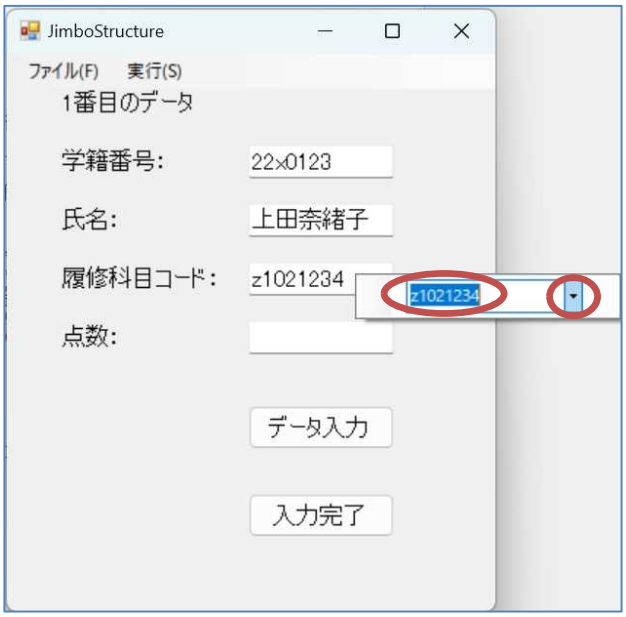
イベントハンドラ toolStripComboBox1_SelectedIndexChanged の処理（赤枠の部分）を記述する。これにより，textBox3 を右クリックして出てくるプルダウンメニューで履修科目コードを選択すると，それを textBox3 に入力することが可能となる。

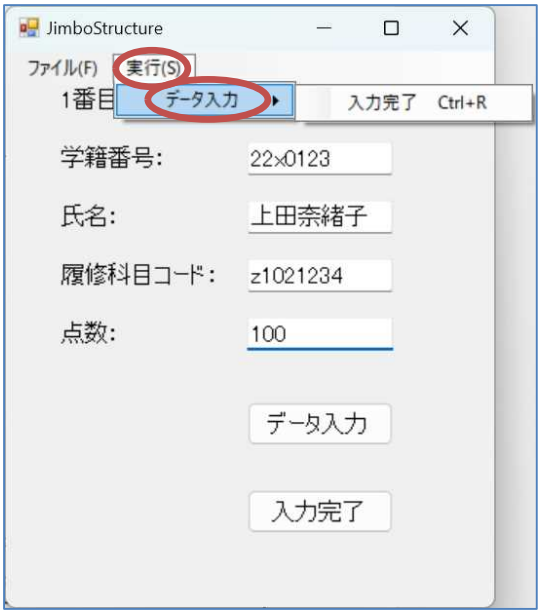
```
        1 個の参照
198     private void toolStripComboBox1_SelectedIndexChanged(object sender, EventArgs e)
199     {
200         textBox3.Text = toolStripComboBox1.Text;
201     }
```

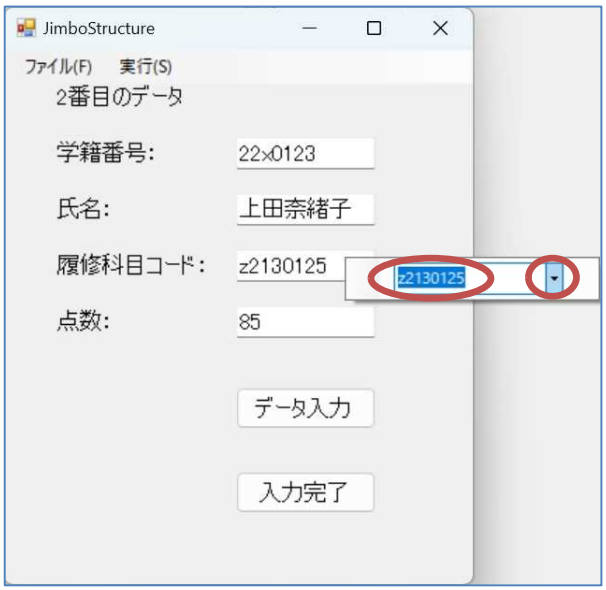
3) 実行

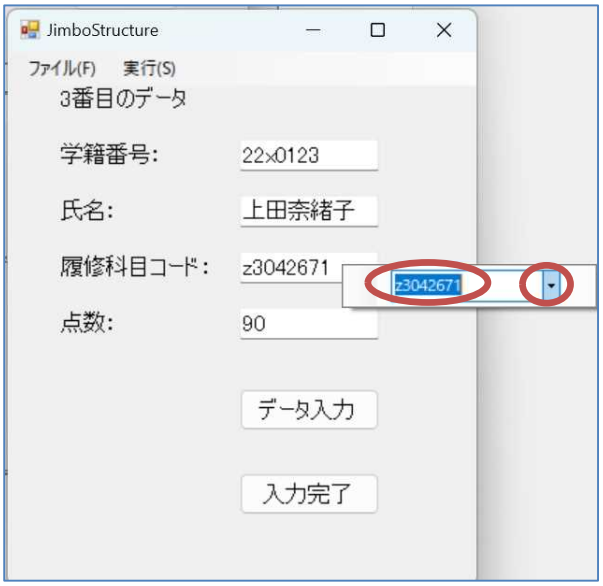
ここで，『保存』ボタンを押してから，『開始』ボタンを押して，プログラムを実行する。

先ず，1 人の学生のデータをテキストボックスに入力して，『実行』メニューにある『データ入力』ボタンをクリックするという操作を 3 件分行う。履修科目コードの入力に際しては，テキストボックスを右クリックして出てくるプルダウンメニューで履修科目コードを選択するという入力支援を利用する。





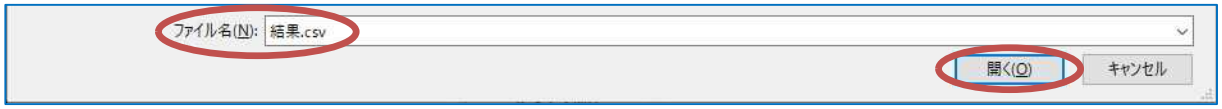




3 件分のデータ入力が完了したら，Ctrl キーを押さえながら R キーを打つ。

［ファイル］→［名前を付けて保存］と選択して，出てくるダイアログで，場所はドキュメントフォルダ，ファイル名は「結果」（鍵括弧は付けない）として保存ボタンをクリックする。

［ファイル］→［開く］と選択して，出てくるダイアログで，場所はドキュメントフォルダ，ファイル名は「結果.csv」（鍵括弧は付けない）として開くボタンをクリックする。



確認を終えたら，Alt キーを押さえながら F4 キーを打って，プログラムを終了する。

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）コードエディタで編集したソースファイル Form1.cs をメールに添付して提出する。

3）完成後に実行して，アプリが起動後，1 件目のデータをテキストボックスに文字入力している途中で履修科目コードの入力支援を利用している様子のスクリーンショット（p.10 の実行結果の最初の図）第 13 回実行結果 1.jpg（.png も可）をメールに添付して提出する。

4）上の状態の後，3 件目のデータ入力が完了し，Ctrl キーを押さえながら R キーを打った状態のスクリーンショット（p.11 の実行結果の最初の図）第 13 回実行結果 2.jpg（.png も可）をメールに添付して提出する。

5）上の状態の後，名前を付けて保存したファイルを開いた際に表示される結果のスクリーンショット（p.11 の実行結果の最後の図）第 13 回実行結果 3.jpg（.png も可）をメールに添付して提出する。

6）アプリから名前を付けて保存したファイル結果.csv 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 Prog2_Questions_13th.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。