1. オブジェクト指向は，特にユーザインタフェース（GUI）に関するプログラミングにおいて有効である。下記の例を実行した後，これの他にマニュアル等を参考にして，自分なりの簡単な Windows アプリケーションを作成してみよ。なお，［新しいプロジェクト］で［Windows フォーム アプリケーション］を選択すれば GUI 上での画面設計が可能になるが，ここでは学習のためにすべて C#コードで作成する。

```
using System;
// Windowsのための追加のライブラリが必要である。［ソリューション エクスプローラー］で
// プロジェクト名の［参照設定］を右クリックし，開いたメニューの［参照の追加］を選択する。
// ダイアログが開くので，下記のライブラリにチェック印をつけて，［OK］ボタンを押す。
using System.Windows.Forms;
using System.Drawing;

class Program
{
    [STAThread] // ← Windowsフォームアプリケーションで必要
    public static void Main()
    {
        // メインウィンドウのオブジェクトを生成して実行開始する
        Application.Run(new MainForm());
    }
}

// メインウィンドウはFormクラスを継承して定義する
// コンソールウィンドウが開かないようにするには，Alt+F7キーでプロジェクトのプロパティを開き，
// 左の欄で［アプリケーション］を選び，［出力の種類］を［Windowsアプリケーション］に変更する
class MainForm : Form
{
    // GUI部品（コントロール）はオブジェクトとして実装されている
    private Label label = new Label(), label2 = new Label();
    private Button button = new Button();
    private PictureBox picture = new PictureBox();
    private OpenFileDialog fileDialog = new OpenFileDialog();

    // メインウィンドウのコンストラクタ
    public MainForm()
    {
        // メインウィンドウのサイズと色の設定
        this.Width = 1000;
        this.Height = 500;
        this.BackColor = Color.PaleGreen;

        // フォーム上に文字列を配置するにはラベルを使う
        label.Text = "簡単な画像表示";
        label.Font = new Font("メイリオ", 30);
        label.AutoSize = true;
        // フォーム上にラベルを載せる
        label.Parent = this;

        // フォーム上の座標を指定してボタンを配置する
        button.Top = 200;
        button.Left = 50;
        button.Text = "画像ファイルを開く";
        button.Font = new Font("メイリオ", 20);
        button.AutoSize = true;
        button.Parent = this;
```

```
        // ボタンがクリックされたときの処理を登録する
        //   これは書き方自体は簡単だが，授業では扱わない「デリゲート」（EventHandler）や
        //   「演算子オーバーロード」（+=の再定義）という発展的な文法事項を使用している
        button.Click += new EventHandler(button_Click);

        // ２つ目のラベル（ファイル名表示用）
        label2.Font = new Font("メイリオ", 14);
        label2.AutoSize = true;
        label2.Top = 20;
        label2.Left = 400;
        label2.Hide();
        label2.Parent = this;

        // 静止画像を配置するにはピクチャーボックスを使う
        picture.Top = 60;
        picture.Left = 400;
        picture.Width = 500;
        picture.Height = 350;
        picture.Hide();
        picture.Parent = this;
    }

    // ボタンがクリックされたときの処理
    public void button_Click(Object sender, EventArgs e)
    {
        // ファイルオープンダイアログを表示する
        DialogResult result = fileDialog.ShowDialog();
        if (result == DialogResult.OK)
        {
            // ラベルを使ってファイル名を表示する
            label2.Text = fileDialog.FileName;
            label2.Show();

            // ファイル名を指定して画像を読み込む
            picture.Load(label2.Text);
            picture.Show();
        }
    }

    // オーバーライドによってウィンドウを閉じる処理を標準から変更する
    protected override void OnFormClosing(FormClosingEventArgs e)
    {
        // 基本クラスのメソッドを呼ぶことで，標準の処理を実行する
        base.OnFormClosing(e);

        // Windows終了時には終了確認をしない
        if (e.CloseReason == CloseReason.WindowsShutDown)
            return;

        // メッセージボックスを表示して本当に終了するか確認する
        DialogResult result = MessageBox.Show(this, "本当に終了しますか？",
                                    "プログラムの終了", MessageBoxButtons.YesNo);
        if (result == DialogResult.No)
            e.Cancel = true;
    }
}
```

※ 続きは「ウィンドウプログラミング」で扱います。