トランプゲームを作ってみよう

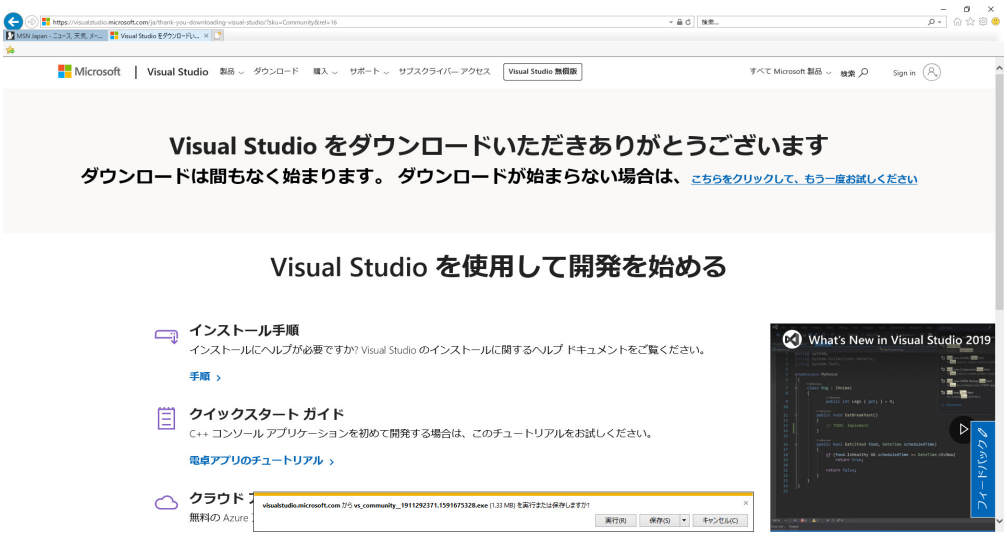
文責　：　高知大学名誉教授　中村　治

ここでは二人でプレイするカード・ゲーム「ジャーマン・ホイスト」を作ってみます。あらかじめお断りしておきますが、何も考えず、書いてある通りプログラムを打ち込んでいくと私が犯したと同じ間違いをすることになります。40 年近くプログラミングを趣味でしていてもいまだに間違いをいっぱいします。デバッグが嫌でプログラミングを断念する人が多いですが、デバッギングをパズルやゲームだと思って楽しんでください。その先に楽しみと深みが広がっています。

パソコンが急に壊れ、新しく購入したパソコンに **Microsoft visual studio 2012 professional** をインストールしてみましたが、うまくいきませんでした。**Microsoft visual studio 2019 community** をデフォルトのままインストールすると「猫でもわかる **Windows** プログラミング」で解説されているような昔の「使って天国、作って地獄」の時代のプログラミングしかできません。しかし、**Microsoft visual studio 2019 community** でも、以下のプログラムを作成できる方法をインターネットで見つけました。こんな抜道を見つけ出す人たちはどんな勉強をしてきたんでしょうか？教育に携わってきた人間として興味があります。

すべてを書き直すのは大変ですから、最低限の修正で済ませることにします。

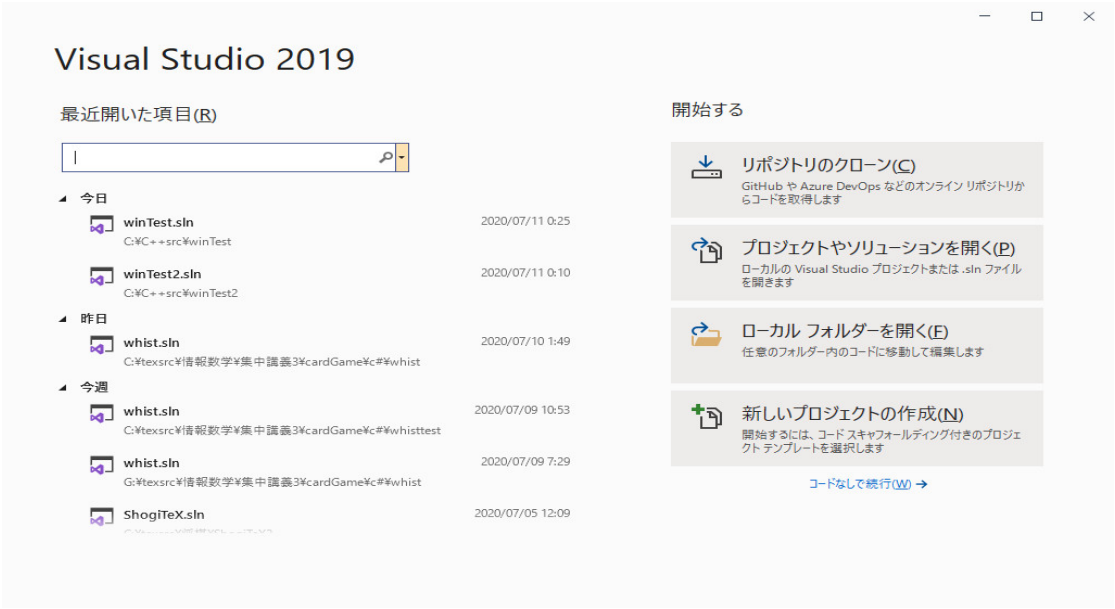
まず Visual Studio の公式サイトから、Community エディションのインストーラーをダウンロードします。



実行をクリックして、インストールを始めます。途中の

の画面のページになったとき、**デフォルトではなく、デスクトップとモバイル（５）の「.NET デスクトップ開発」と「C++によるデスクトップ開発」と右側のオプションの「v142 ビルドツール用 C++/CU サポート」にチェックを入れることを忘れないでください。**後はデフォルトでいいです。

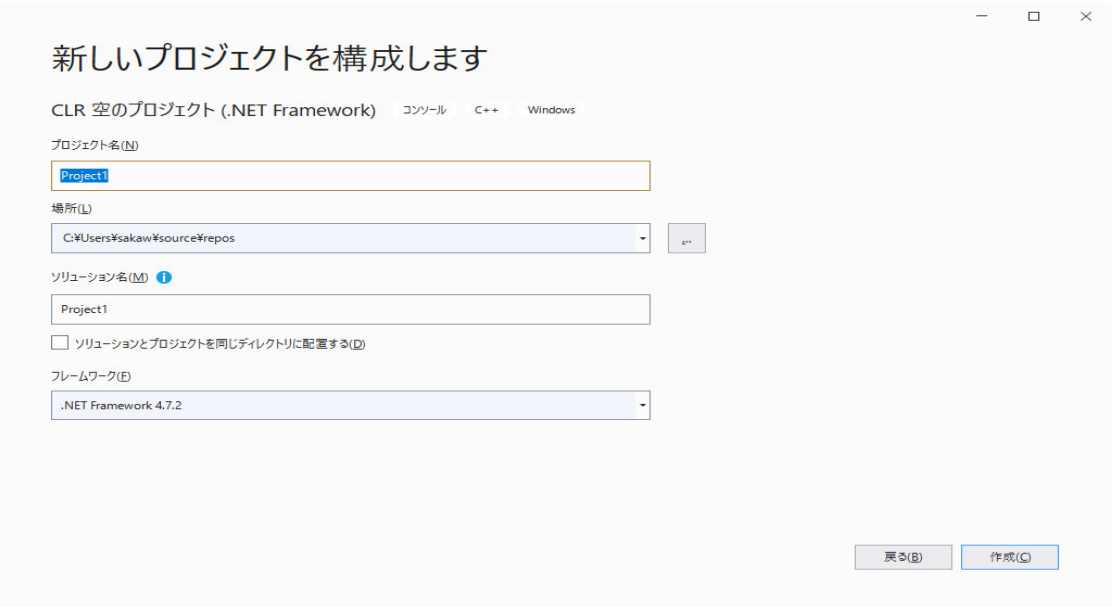
まず、Microsoft Visual Studio 2019 を立ち上げる。



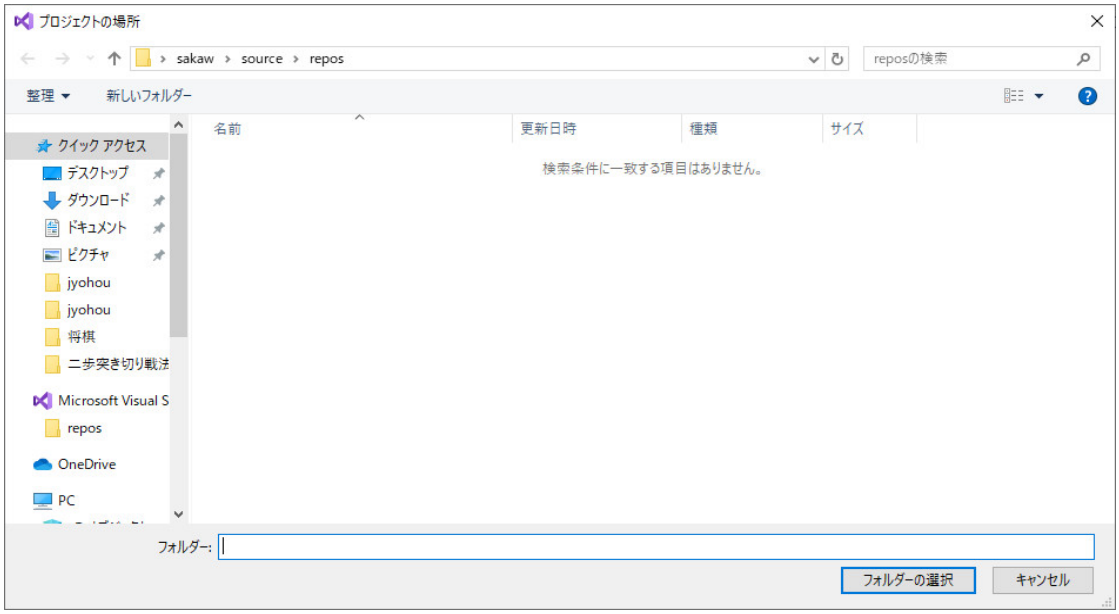
私は既に使っていて、左側にその履歴が残っていますが、それは気にしなくていいです。右の一番下の「新しいプロジェクトの作成（N)」をクリックします。

この画面で、右側の選択肢の下の方にある「CLR 空のプロジェクト（.NET Framework)」を選択し、

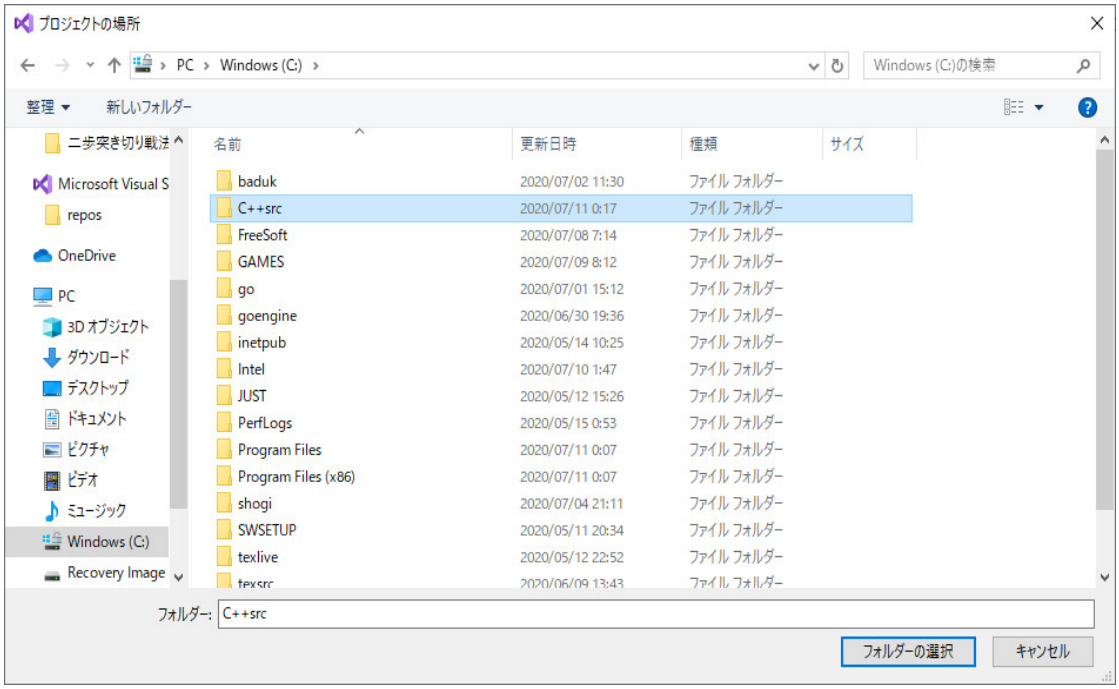
「次へ（N)」のボタンをクリックします。



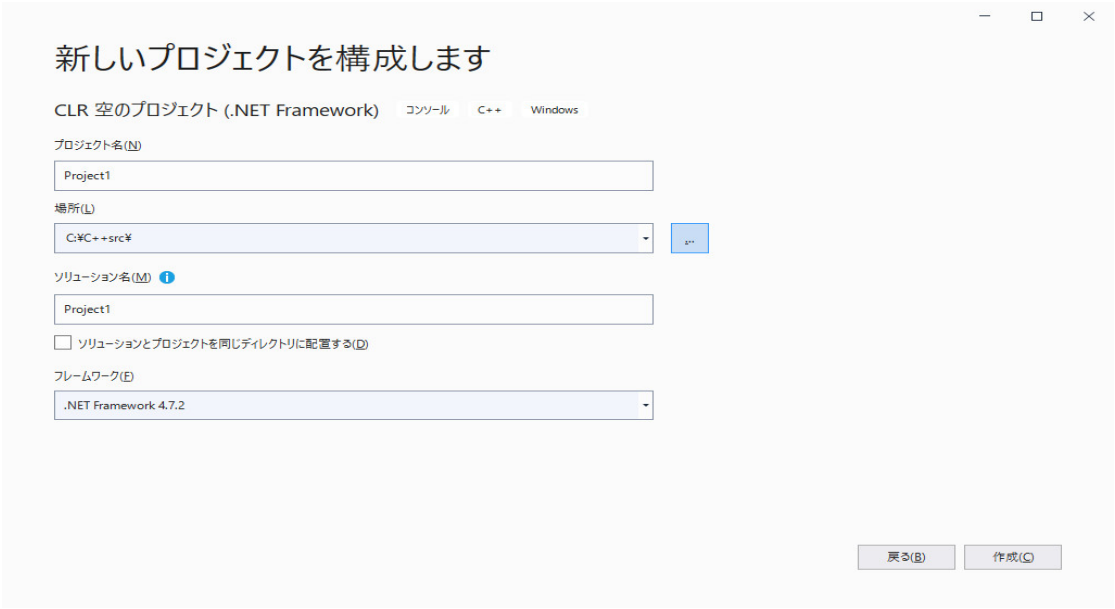
「場所（L)」の表示が異なっていると思いますが気にしなくてもいいです。どこにプロジェクト（作るプログラム）を保存するか気にしなければ、デフォルトでもいいですが、右側の四角をクリックすると



とプロジェクトを保存するフォルダを指定できます。私は「C++src」というフォルダを作って、C++のプログラムはそこに保存するようにしています。

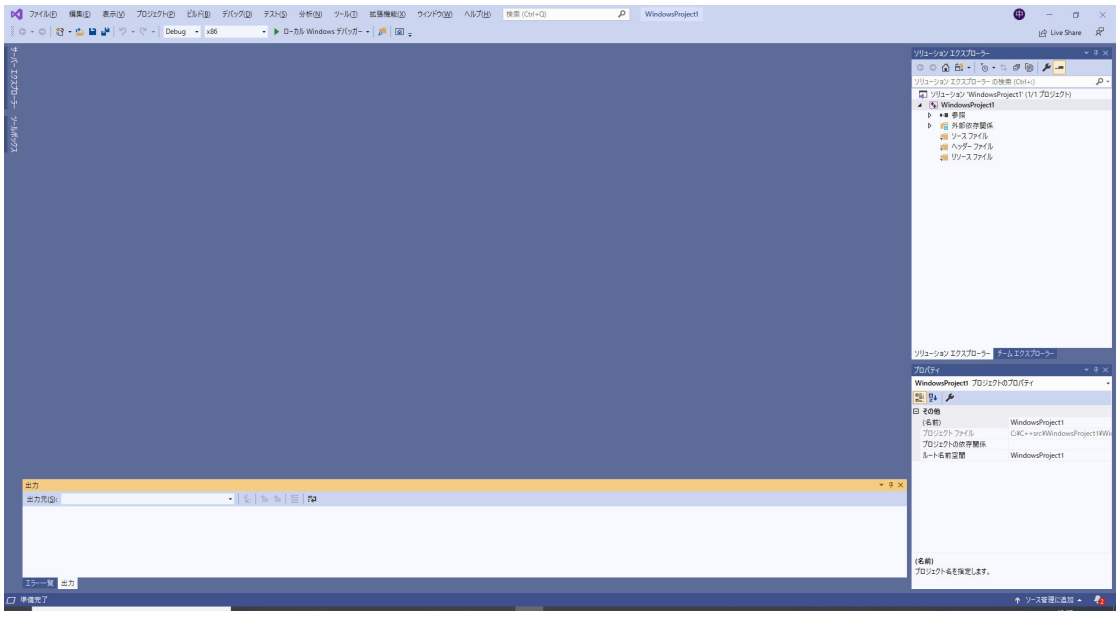


「フォルダーの選択」のボタンをクリックします。



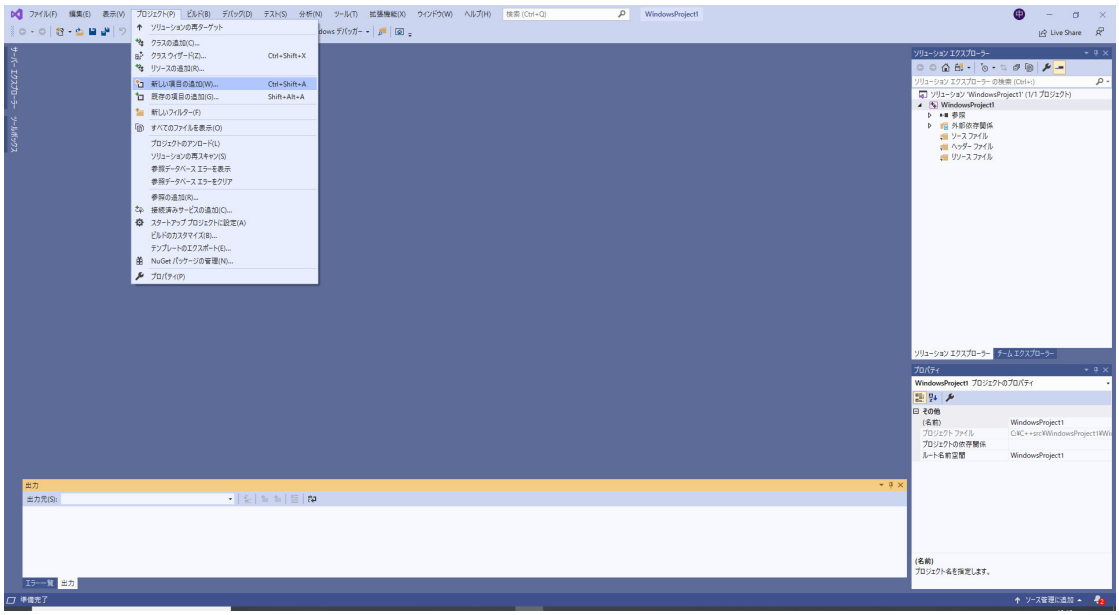
「プロジェクト名（N）」をプログラムの内容を表す名前に変えることが出来ます。ここでは「windowsProject1」としてみます。プロジェクト名を設定すると自動的に同じ名前が「ソリューション名」に設定されます。「ソリューション名」を別の名前に設定できますが、あえて変える必要はありません。「ソリューション名」と同じ名前のフォルダが「場所」に指定された所に作成され、そのフォルダ内に必要なソフトやフォルダが作られます。

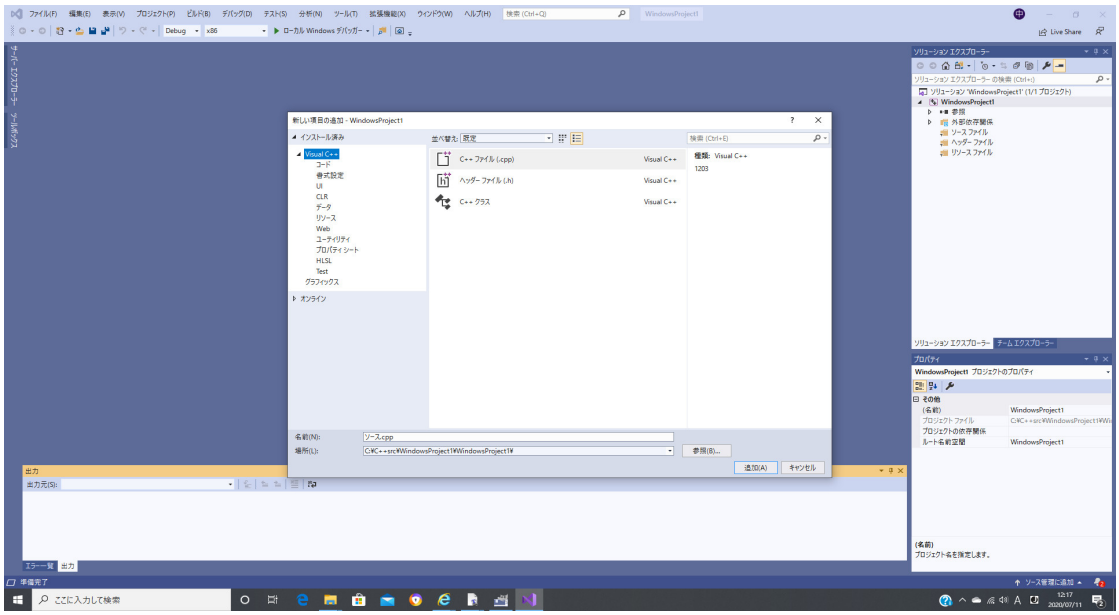
「作成（C）」をクリックします。



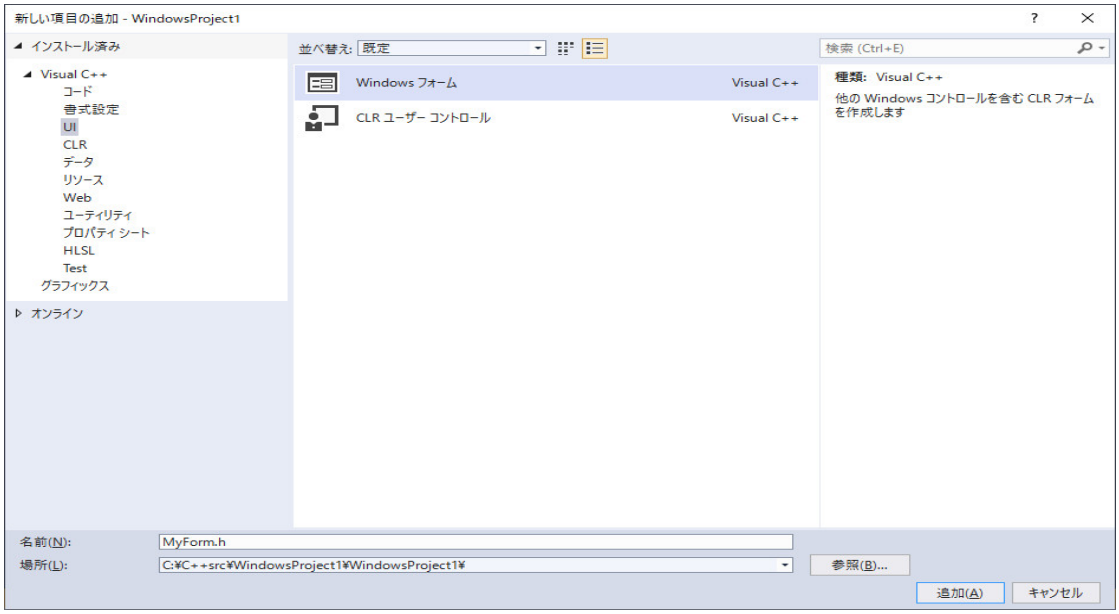
となります。

上段のメニューの「プロジェクト」の「新しい項目の追加」を選択します。

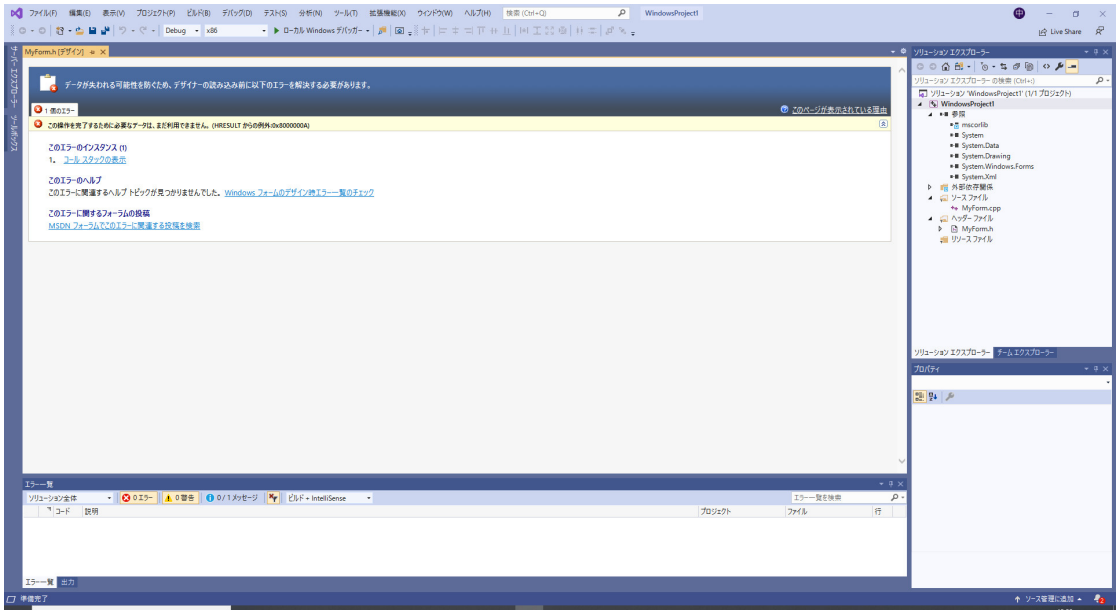




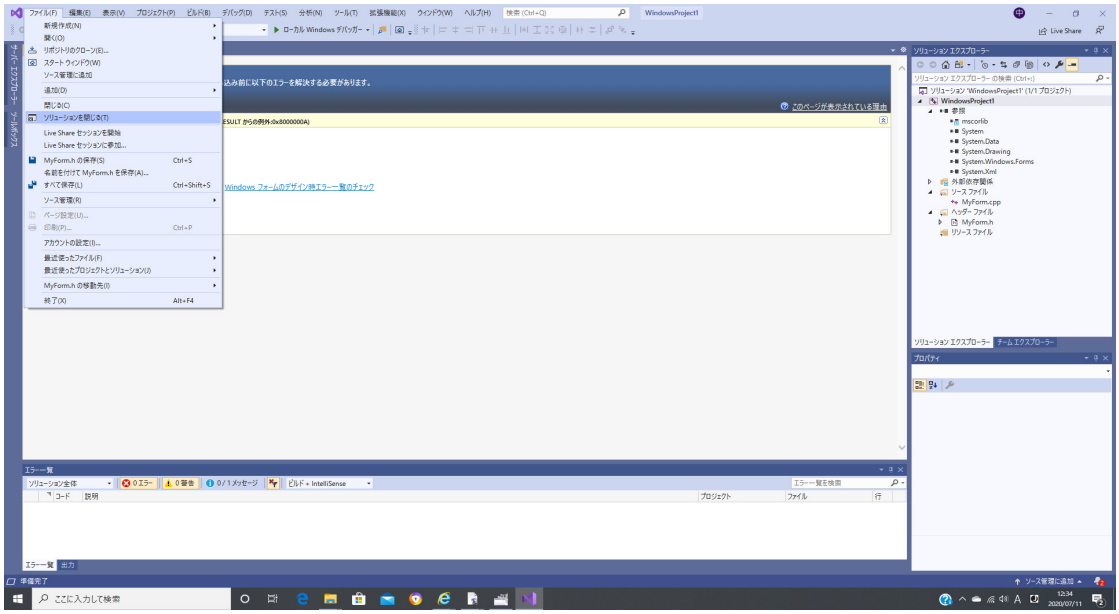
となります。



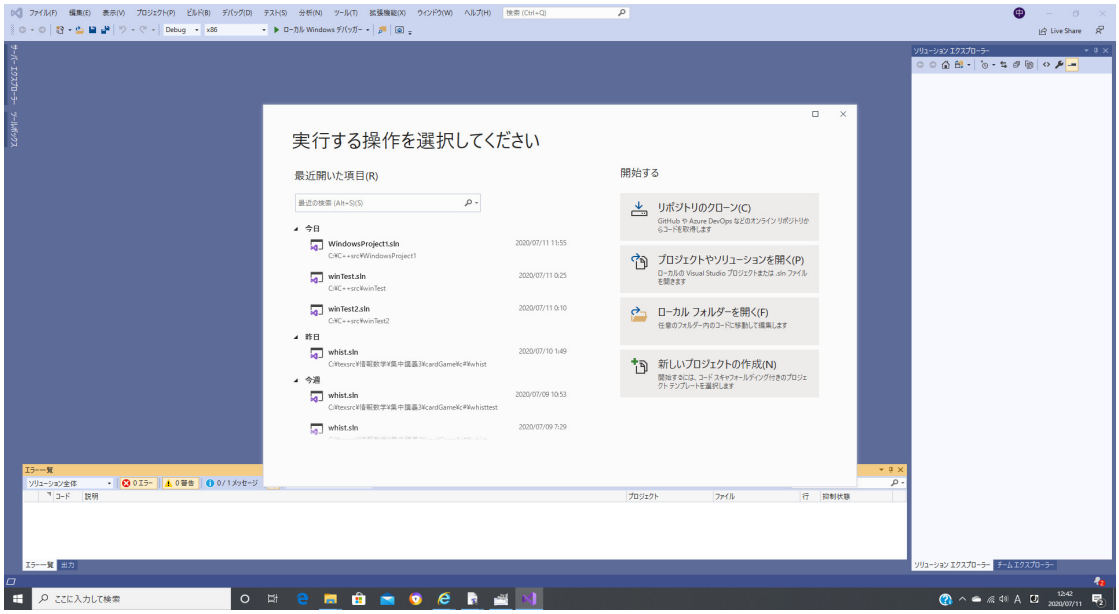
上図のように、「Visual C++」の「UI」の「Windows フォーム」を選択し、「追加（A)」をクリックします。



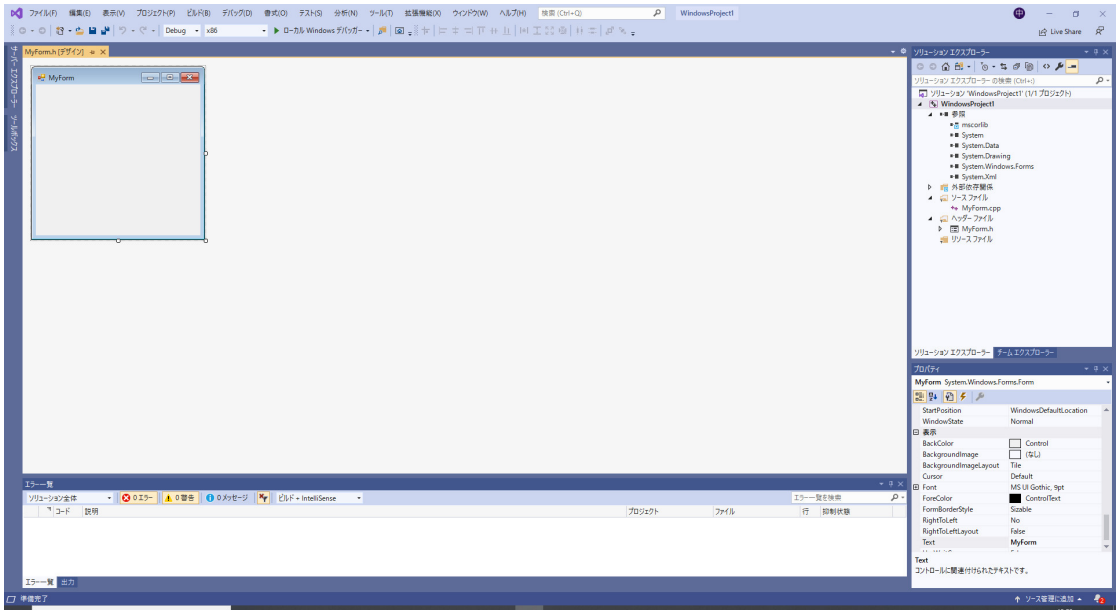
のようなエラーメッセージが表示されます。上段のメニューの「ファイル」の「ソリューションを閉じる（T)」を



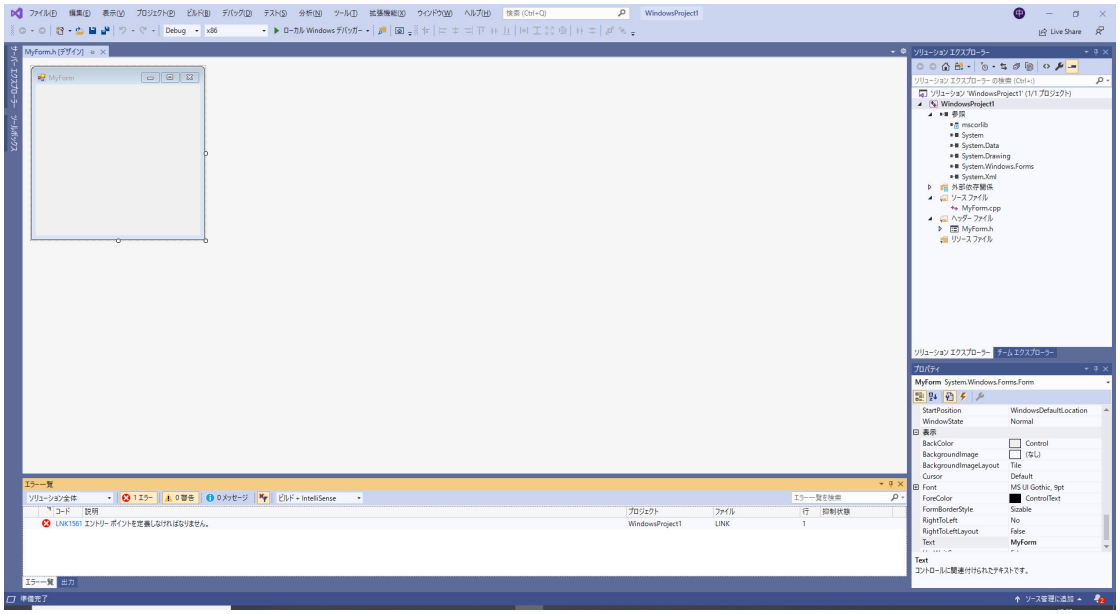
クリックします。



となります。左側の「最近開いた項目（R)」の「今日」の一番上にプロジェクト名の「WindowsProject1.sln」がありますから、それをクリックします。

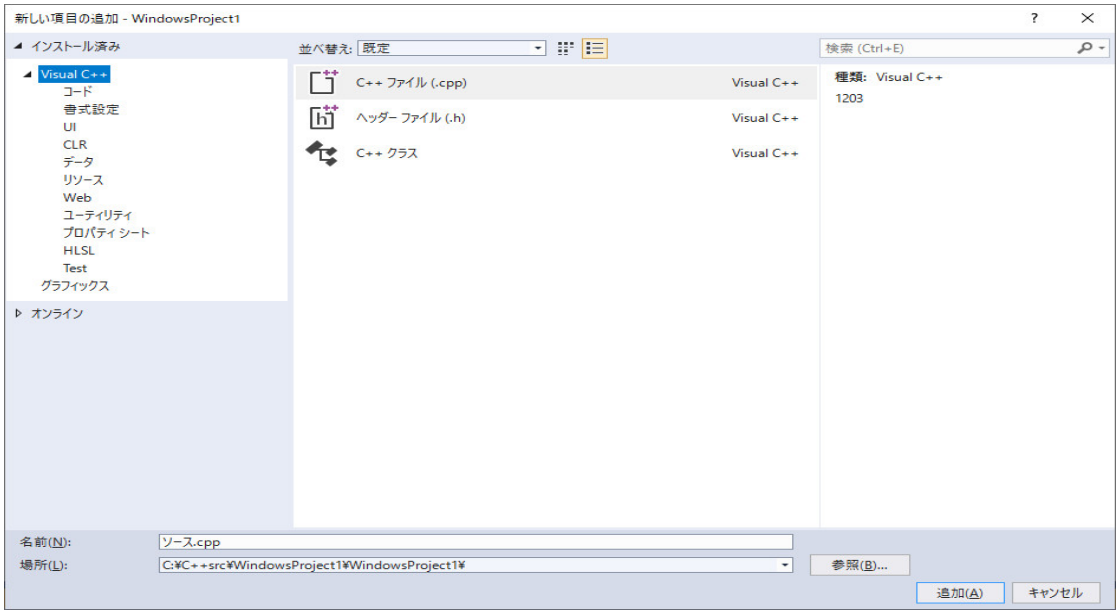


となります。昔の visual studio C++ はこの状態でビルドするとよかったのですが、「ビルド」の「ソリューションのビルド（B)」をクリックすると

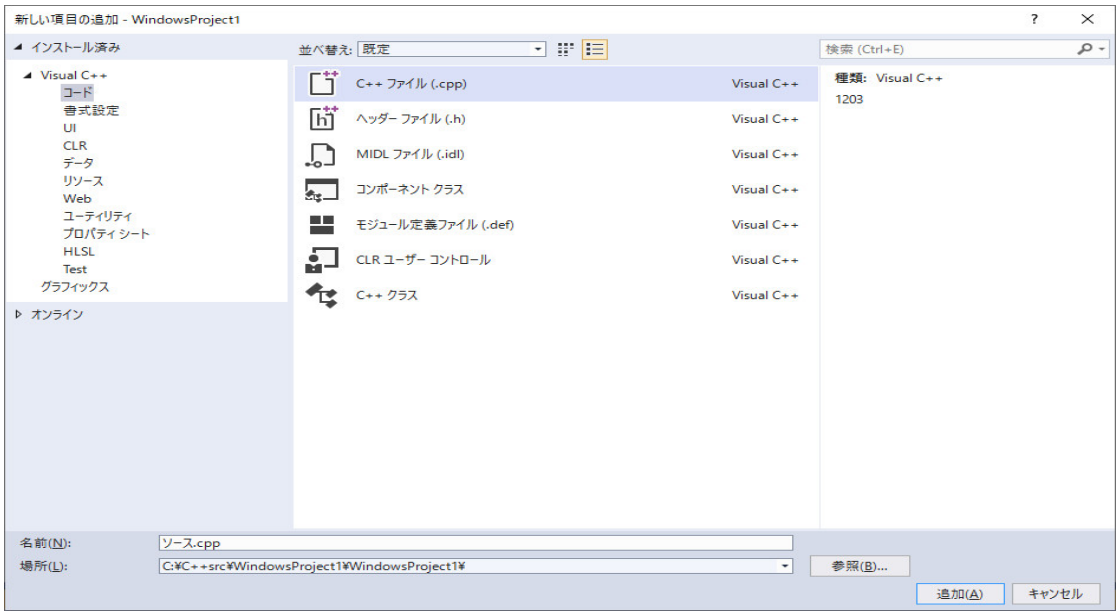


のようにエラーメッセージが表示されます。このフォームを実行時に作成して表示するためのプログラムを用意する必要があります。

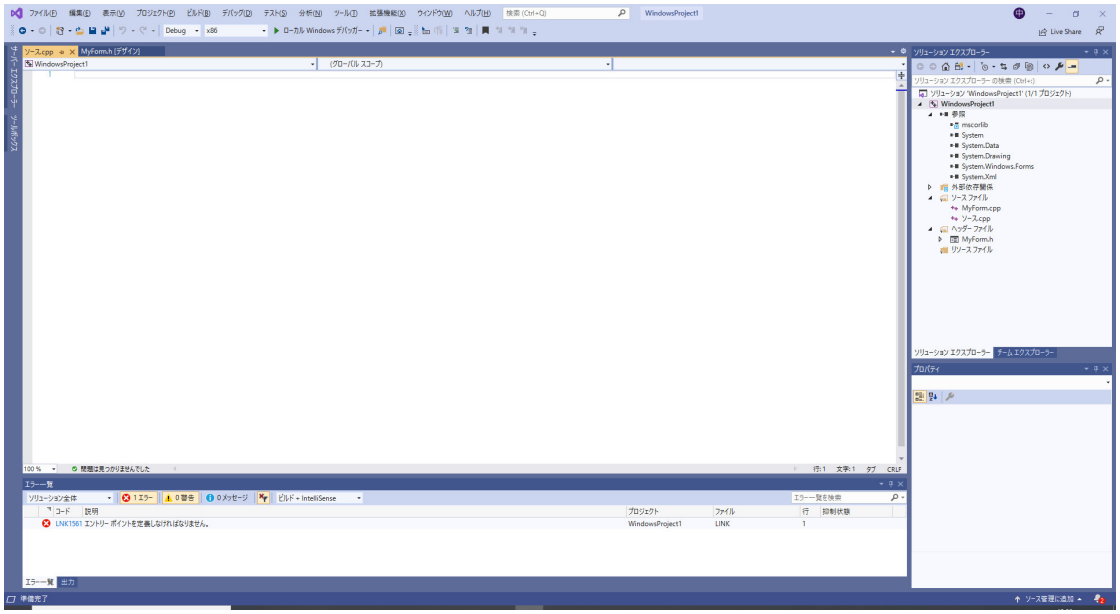
上段のメニューの「プロジェクト」の「新しい項目の追加」を選択します。



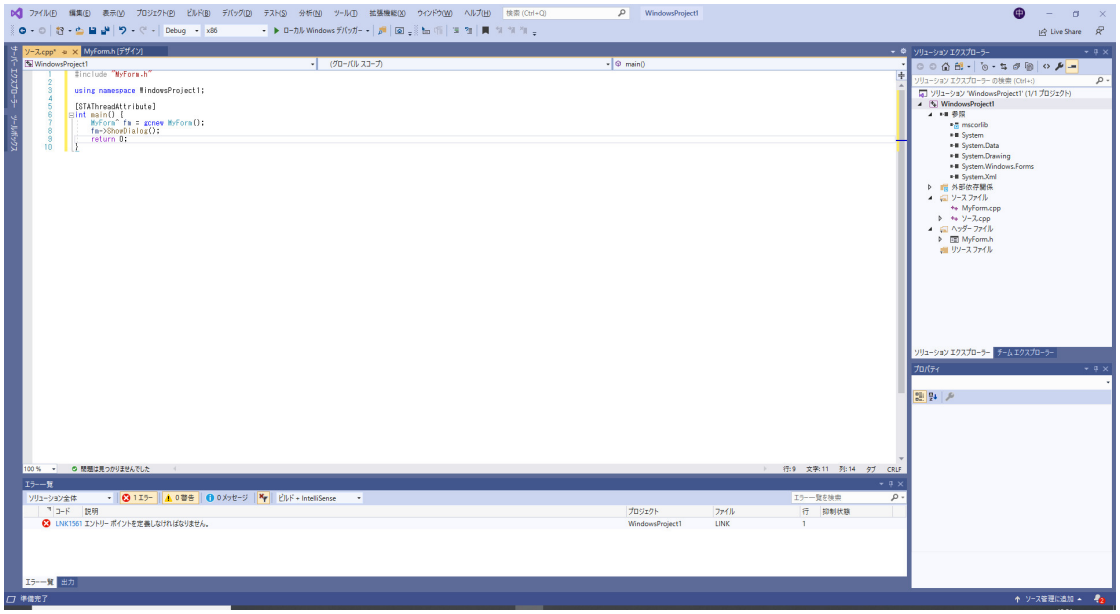
の画面で、今度は、「Visual C++」の「コード」の「C++ファイル（.cpp)」を選択し、



「追加（A)」をクリックします。



と「ソース.cpp」が作られ、何も書いてない空のファイルが表示されます。



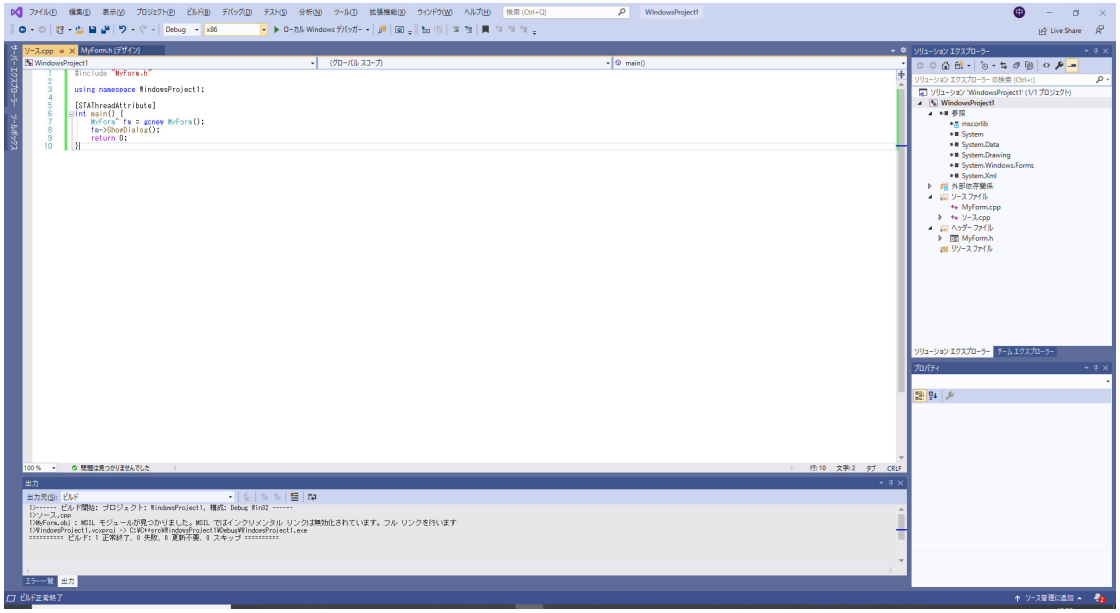
のように

```
#include "MyForm.h"

using namespace WindowsProject1;

[STAThreadAttribute]
int main() {
	MyForm^ fm = gcnew MyForm();
```

```
        fm->ShowDialog();
        return 0;
}
```

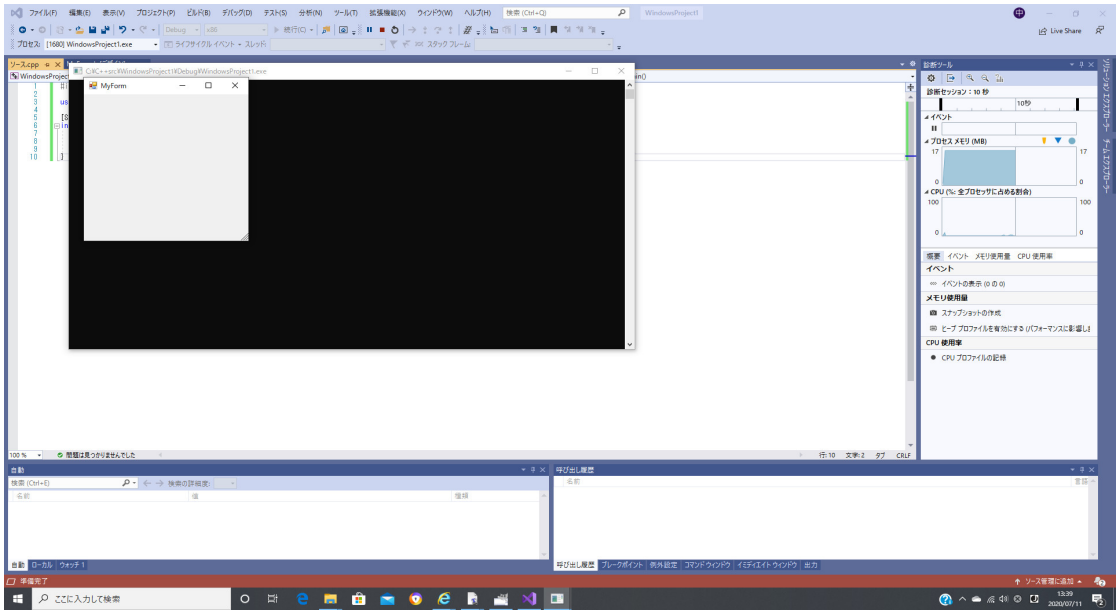
と打ち込みます。

「ビルド」の「ソリューションのビルド（B)」をクリックすると



のように、正常にビルドできます。

「デバッグ」の「デバッグの開始（S)」をクリックすると



のように、フォームが表示されます。黒い画面も一緒に表示されますが気にしなくてもいいです。

随分、手間がかかりましたが、何もないフォームを表示することが出来ました。

これまでの手続きは魔法使いの儀式だと思って、以下には古い visual studio C++ の使い方を解説していますが、今までと通り、フォームに「ツールボックス」にあるツールを配置して、プログラミングをすればいいです。

ジャーマン・ホイストはホイストのバリエーションの一つで、2 人でプレイするように工夫されたものです。松田道弘著「トランプゲーム辞典」東京堂出版に載っています。

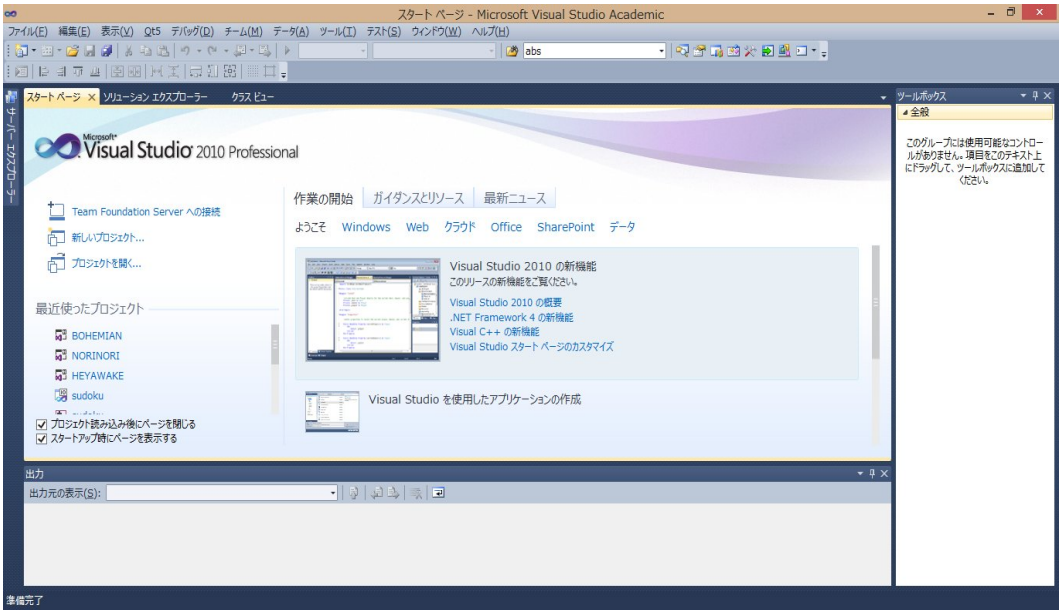
ルールは

1. 人数は 2 人
2. 52 枚のカードを使用する
3. カードの順位は A, K, Q, J, 10, 9, 8, 7, 6, 5, 4, 3, 2 の順です
4. 手札は各人 13 枚で、残りはストックとしてテーブルに置き、ストックのトップカードだけを表向きにします。このカードのスーツが切り札になります。
5. ディーラーでない人がオープニング・リードをします。
6. リードされたスーツは必ず出さなければなりません。 リードされたスーツを持っていなければ、どのカードを出しても良いです。
7. 最強の切り札を出した人か、または切り札が場に出ていなければリードされたスーツの一番高いランクのカードを出した人がトリックを取ります。
8. トリックを取った人は、ストックのトップ・カード（切り札）を取って手札に入れ、相手はその下のカード（裏向き）を取ります。相手には見せません。
9. 各プレーヤーの手札は 13 枚に戻りました。この時、トリックを取った人は、次のリードをする前に、ストックの一番上のカードを表向きにします。
10. 第 1 トリックを取った人がリードして、このトリックの勝者がストックの表向きのカードを取り、相手がその下のカードを取ります。
11. この手順をストックが無くなるまで続け、その後は手札が無くなるまでトリック争奪のプレイを続けます。
12. プレイが終わると取得トリックの数を比べ、勝った人はその差を得点します。トリック数が同数の時は両者無得点です。

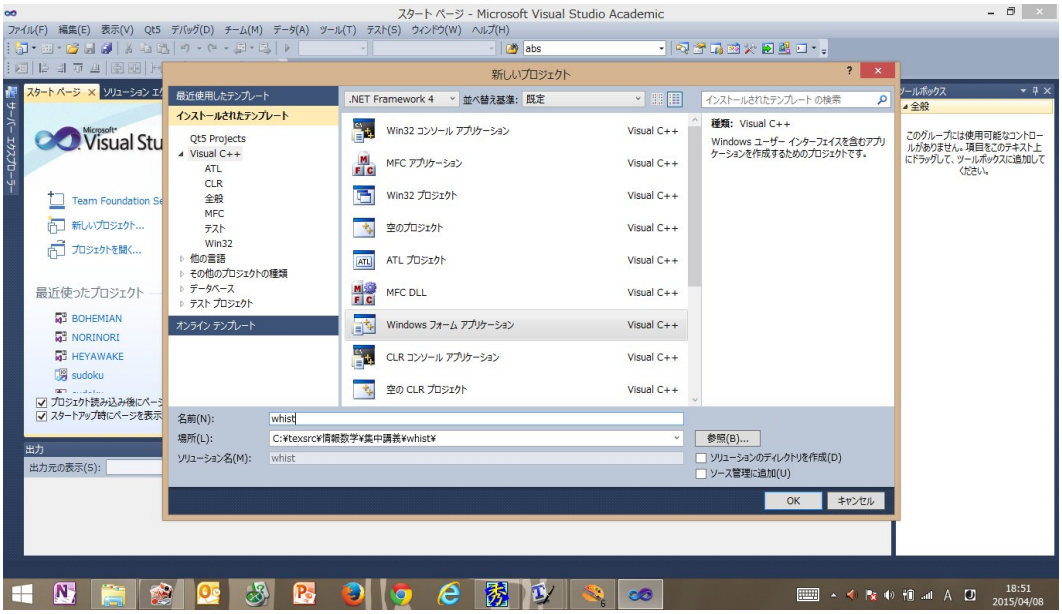
です。

このジャーマン・ホイストのプログラムを VC++で作成し、コンピュータと対戦できるようにします。

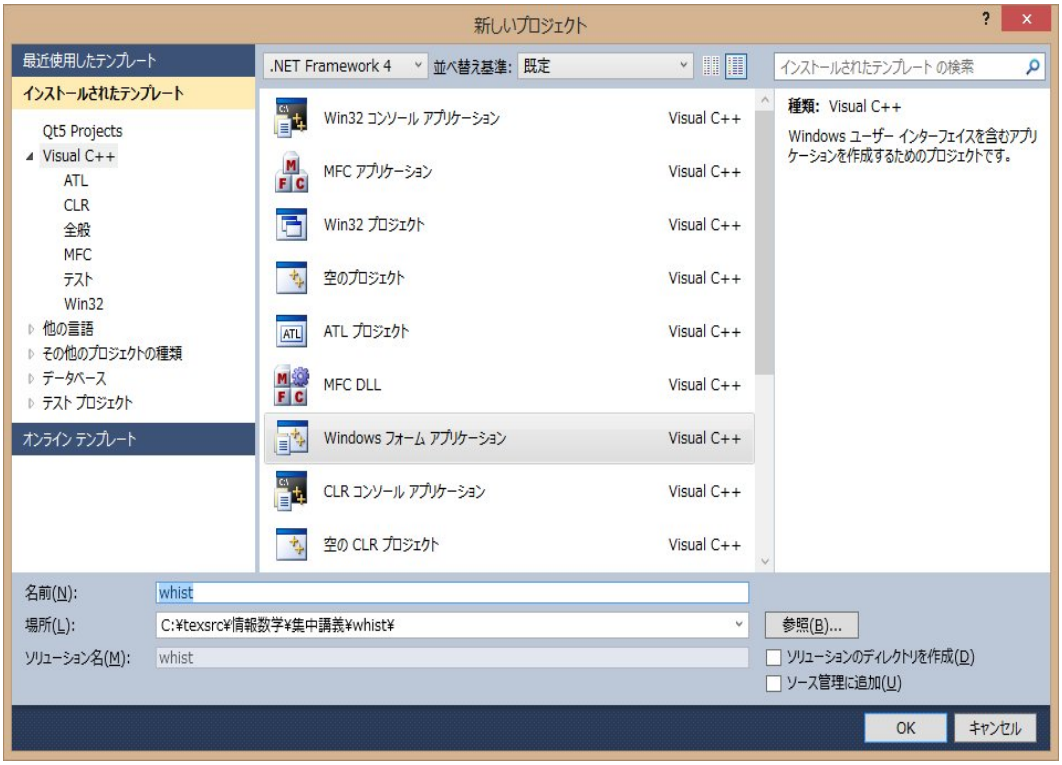
VC++ を立ち上げます。



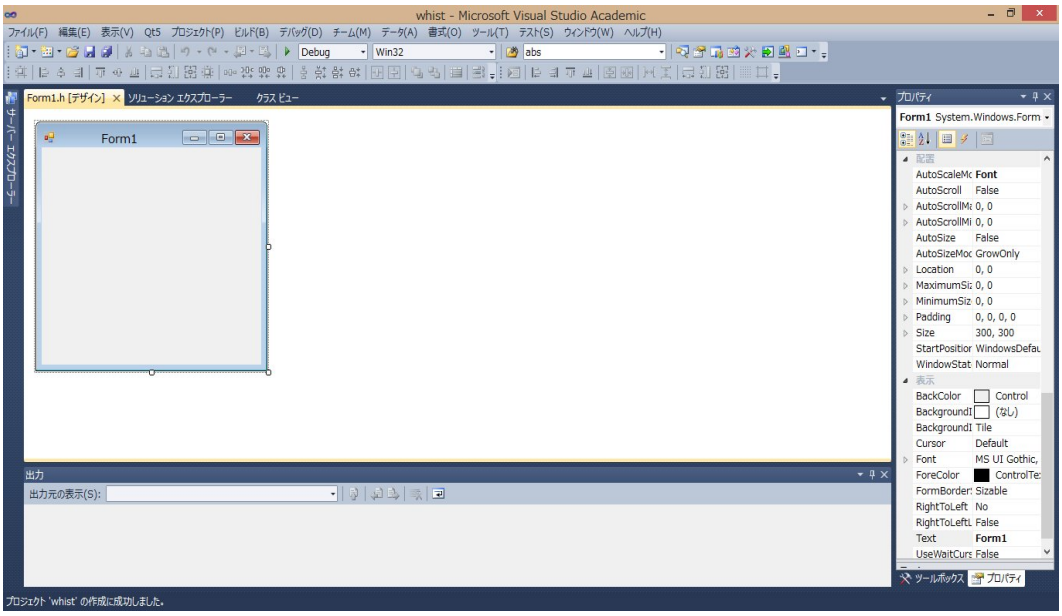
画面は Visual C++ Professional 2010 ですが、Visual C++ 2010 express　で大丈夫です。「新しいプロジェクト」をクリックします。



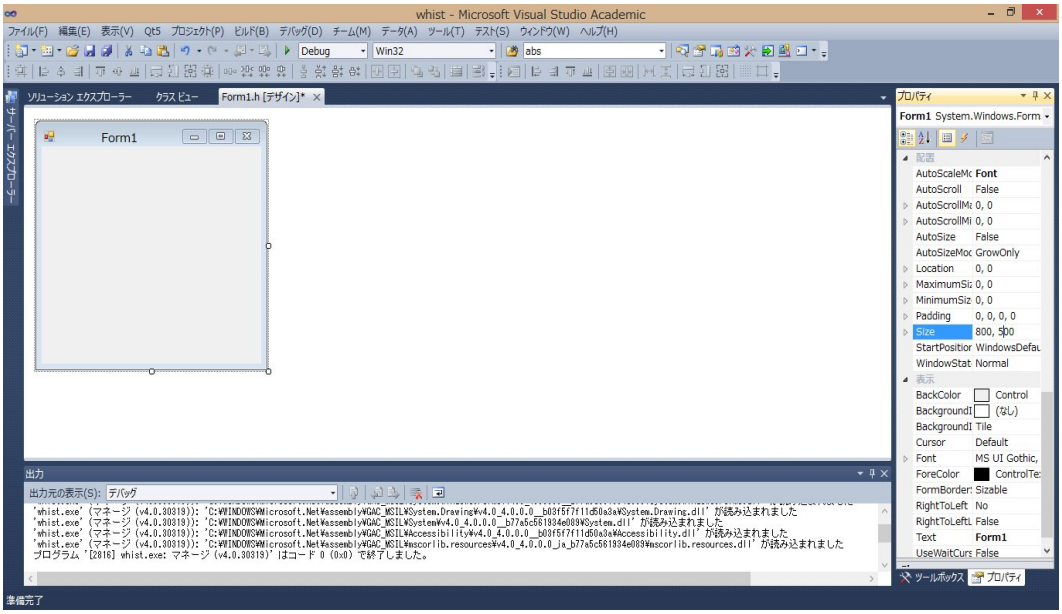
「Windows フォームアプリケーション」を選択し、「名前」のテキストボックスに whist と打ち込みます。



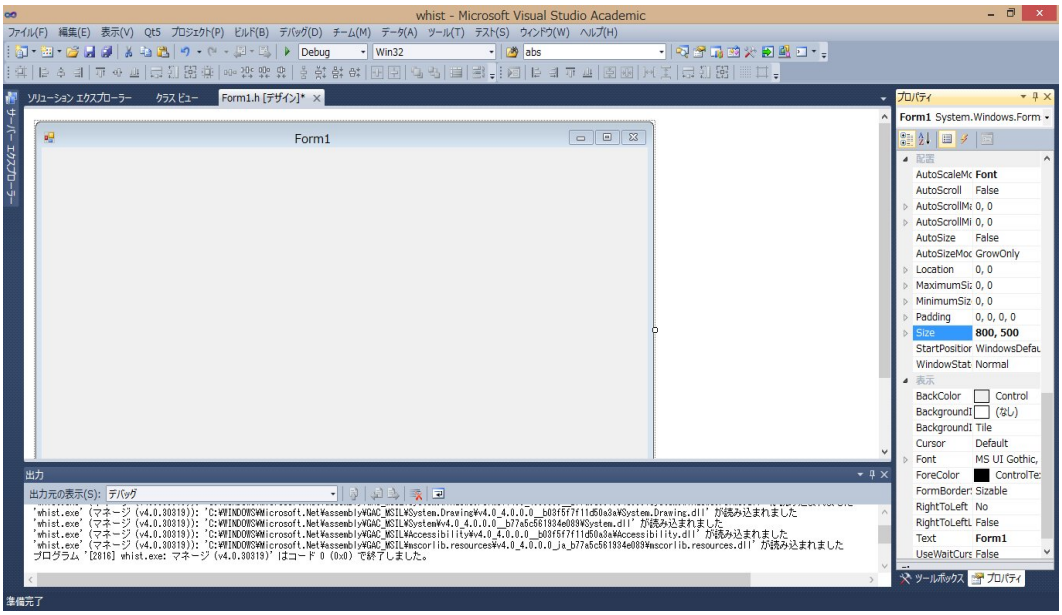
「OK] のボタンをクリックします。



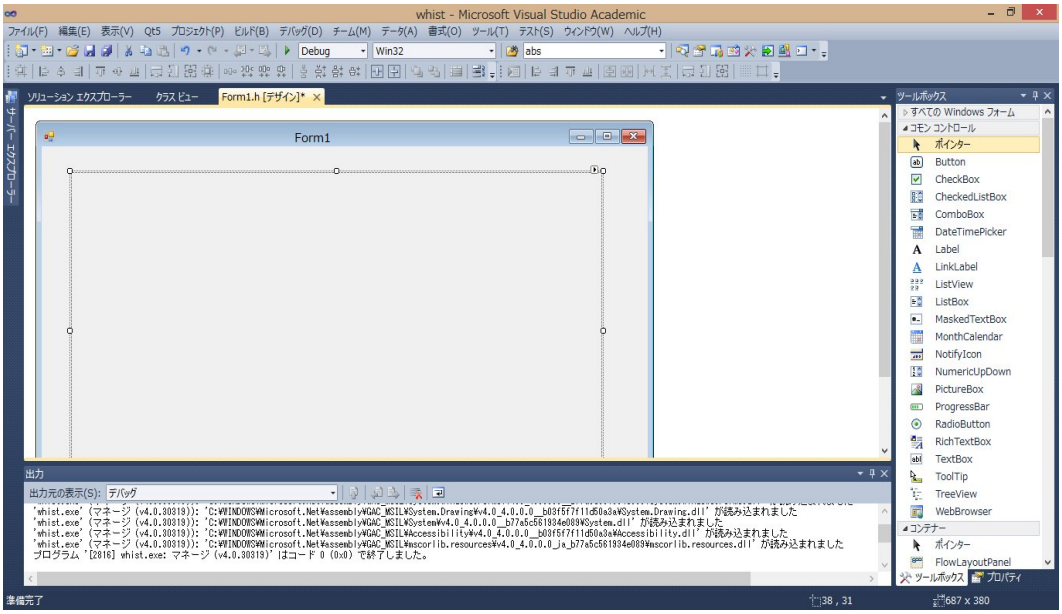
となります。



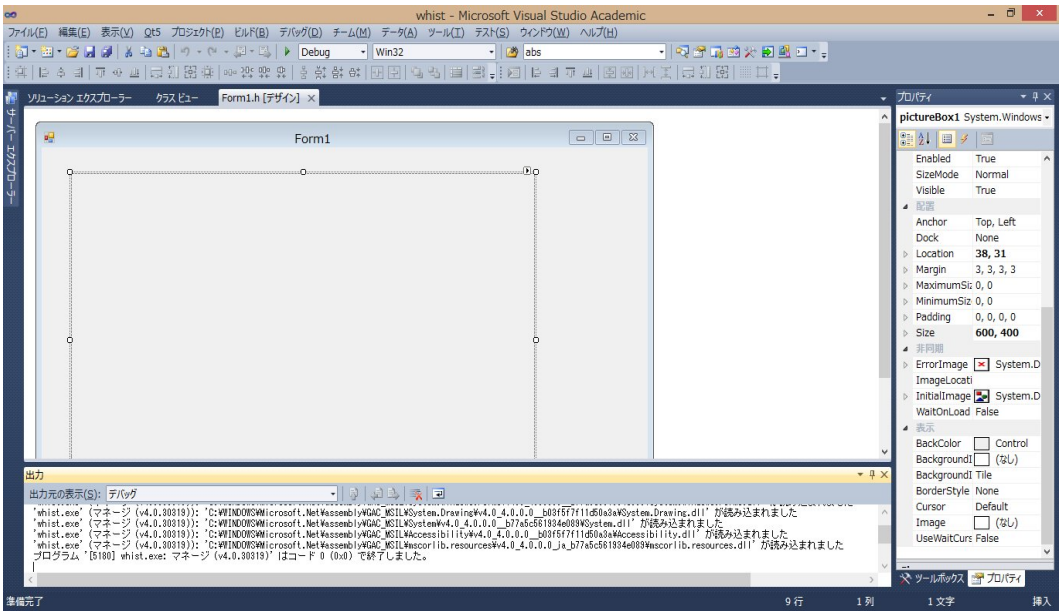
Form1 の size を　800, 500 とします。



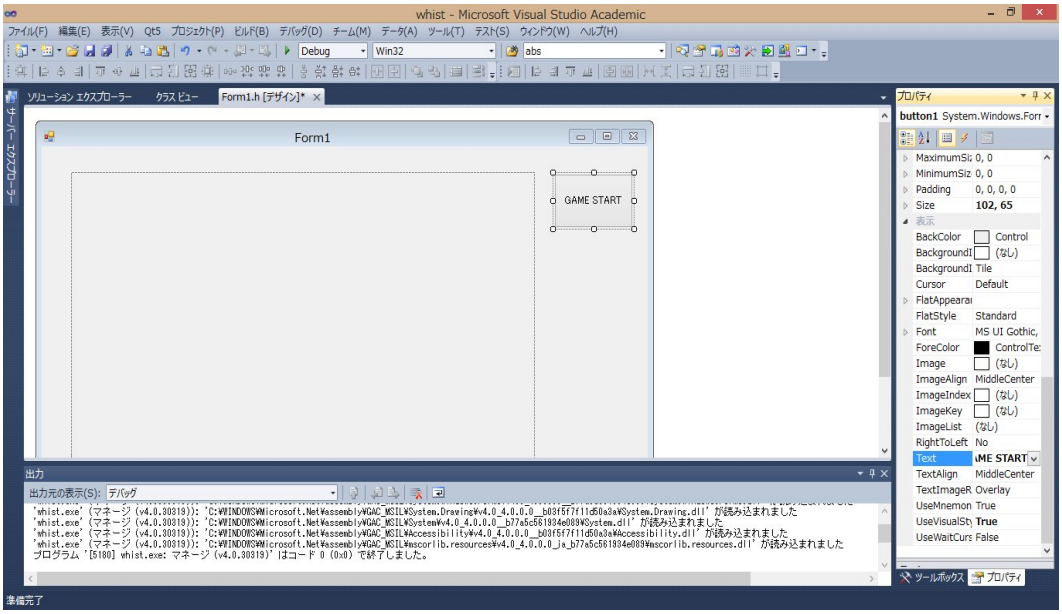
「ツールボックス」の「PictureBox」をクリックし、Form1 でドラッグして「PictureBox」を配置します。



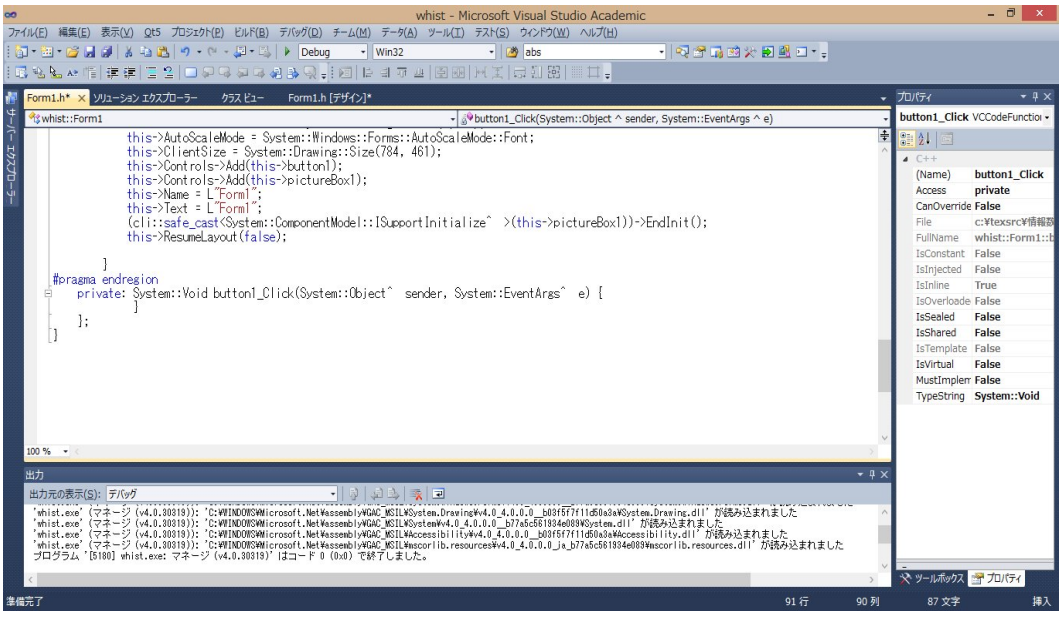
「PictureBox」をクリックし」、「プロパティ」で size を 600, 400 にします。



「ツールボックス」の「button」をクリックし、Form1 でドラッグして「button」を配置します。
「プロパティ」で Text を GAME START にします。



「GAME START」のボタンをダブルクリックします。

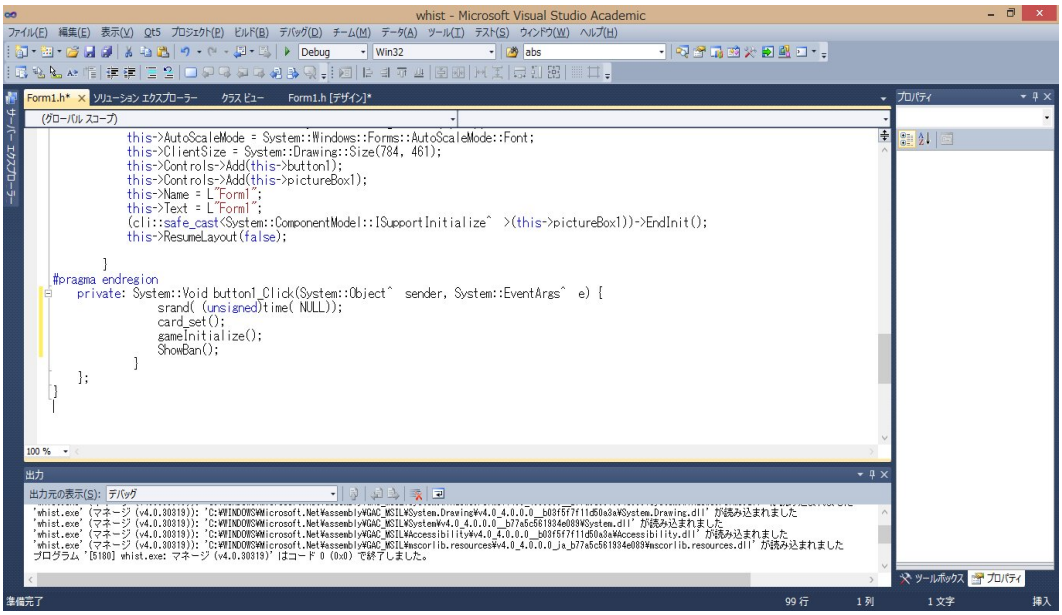



```
private: System::Void button1_Click(System::Object^  sender, System::EventArgs^  e) {
 }
```

にゲームが始まる初期画面を表示するプログラムを書きます。

```
private: System::Void button1_Click(System::Object^  sender, System::EventArgs^  e) {
    srand( (unsigned)time( NULL));
    card_set();
    gameInitialize();
    ShowBan();
}
```

と打ち込みます。

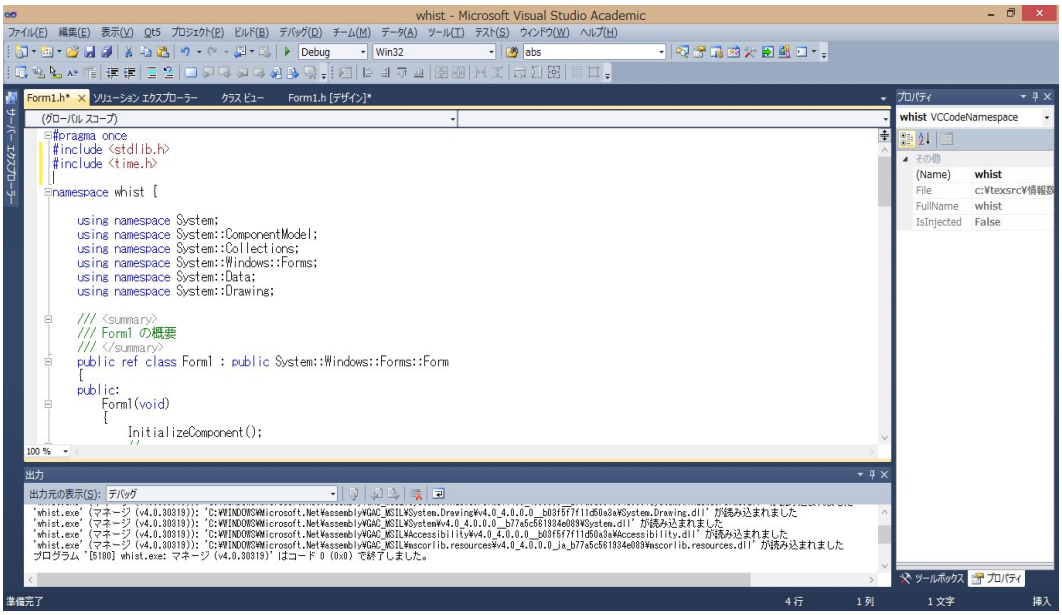



```
srand( (unsigned)time( NULL));
```

は乱数を実行するたびに別の列になるするためのものです。この関数を使うために Form1.h の先頭に

```
#include <stdlib.h>
#include <time.h>
```

と打ち込みます。




```
card_set();
```

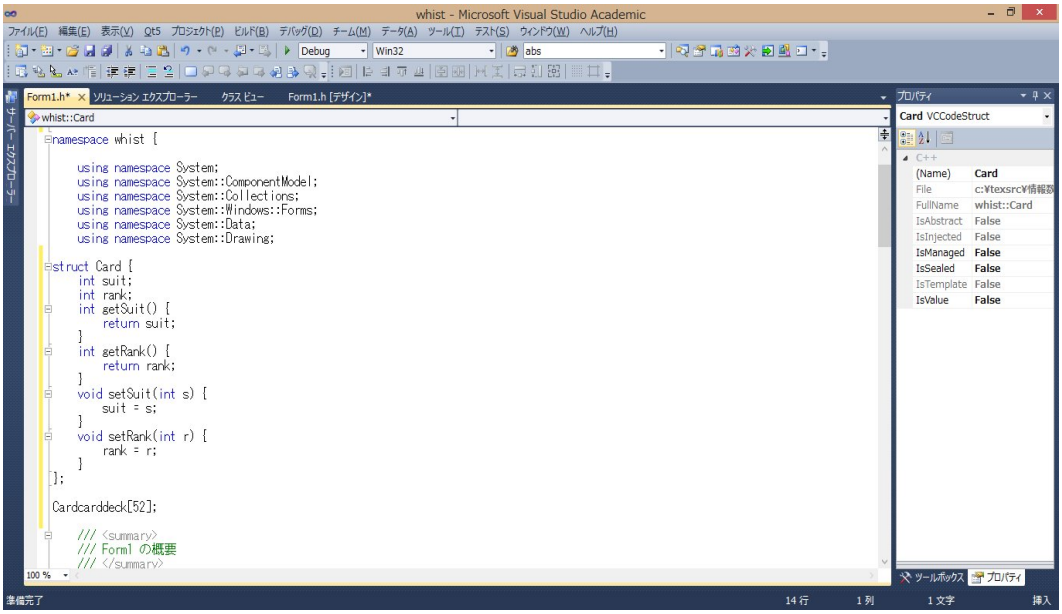
で、一組のカードの準備をします。一つ一つのカードを Card と言うクラスで表しますが、class と言う言葉を使うと上手く行かない（Form1.h「デザイン」が表示できまくなる）ので struct を使います。

```
struct Card {
   int suit;
   int rank;
   int getSuit() {
      return suit;
   }
   int getRank() {
      return rank;
   }
   void setSuit(int s) {
      suit = s;
   }
   void setRank(int r) {
      rank = r;
  }
};

Card carddeck[52];
```

で、カードの為のクラスと一組のカードを表現しています。それぞれのカードはスーツとランクを持っています。この定義を Form1.h の先頭の

```
namespace whist {
    using namespace System;
    using namespace System::ComponentModel;
    using namespace System::Collections;
    using namespace System::Windows::Forms;
    using namespace System::Data;
    using namespace System::Drawing;
```

の後に打ち込みます。



上で定義したクラスを使って、関数

```
card_set();
```

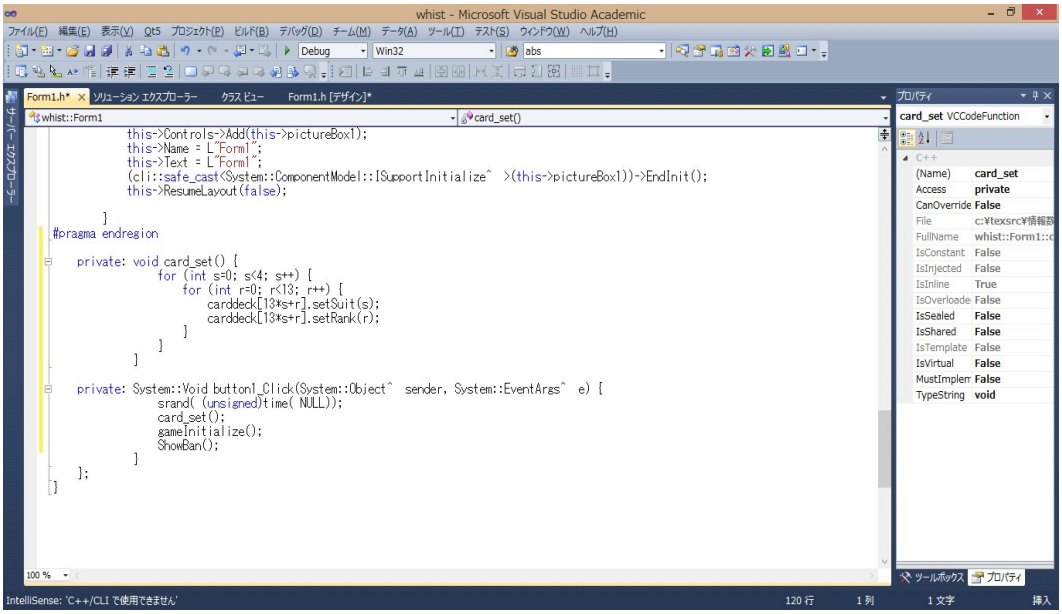
で、一組のカード carddeck[52] を準備します。定義は

```
private: void card_set() {
    for (int s=0; s<4; s++) {
        for (int r=0; r<13; r++) {
            carddeck[13*s+r].setSuit(s);
            carddeck[13*s+r].setRank(r);
        }
    }
}
```

です。これを

```
private: System::Void button1_Click(System::Object^  sender, System::EventArgs^  e) {
    srand( (unsigned)time( NULL));
    card_set();
    gameInitialize();
    ShowBan();
}
```

の直前に打ち込みます。



次にカードをシャッフルし、カードを配ります。そのための関数が

```
gameInitialize();
```

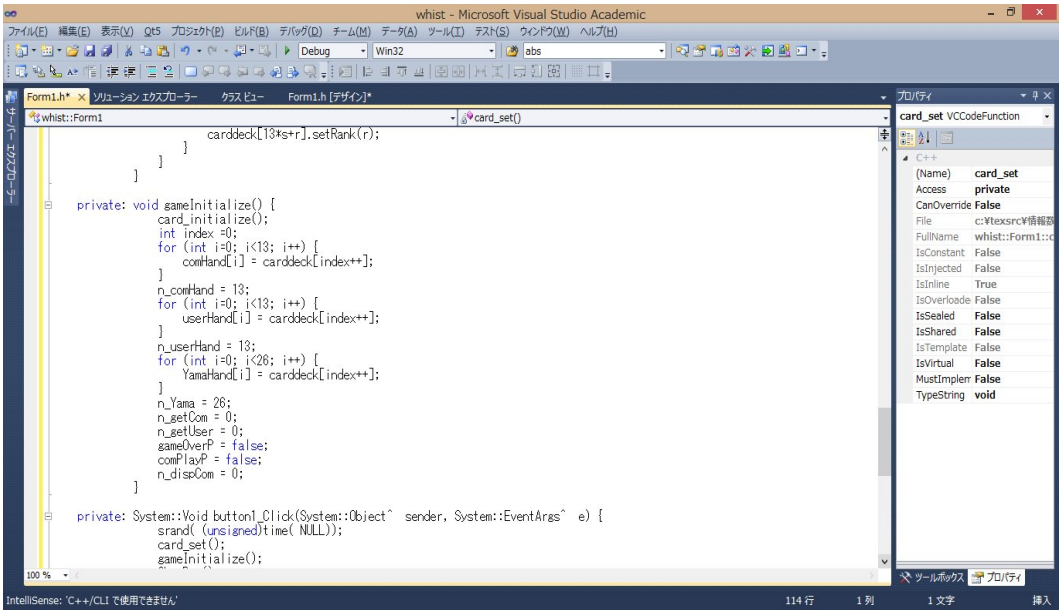
です。定義は

```
private: void gameInitialize() {
    card_initialize();
    int index =0;
    for (int i=0; i<13; i++) {
        comHand[i] = carddeck[index++];
    }
    n_comHand = 13;
    for (int i=0; i<13; i++) {
        userHand[i] = carddeck[index++];
    }
    n_userHand = 13;
    for (int i=0; i<26; i++) {
        YamaHand[i] = carddeck[index++];
    }
    n_Yama = 26;
    n_getCom = 0;
    n_getUser = 0;
    gameOverP = false;
    comPlayP = false;
    n_dispCom = 0;
}
```

です。これを

```
private: System::Void button1_Click(System::Object^  sender, System::EventArgs^  e) {
    srand( (unsigned)time( NULL));
    card_set();
    gameInitialize();
    ShowBan();
}
```

の直前に打ち込みます。




```
    card_initialize();
```

で、カードをシャッフルします。定義は

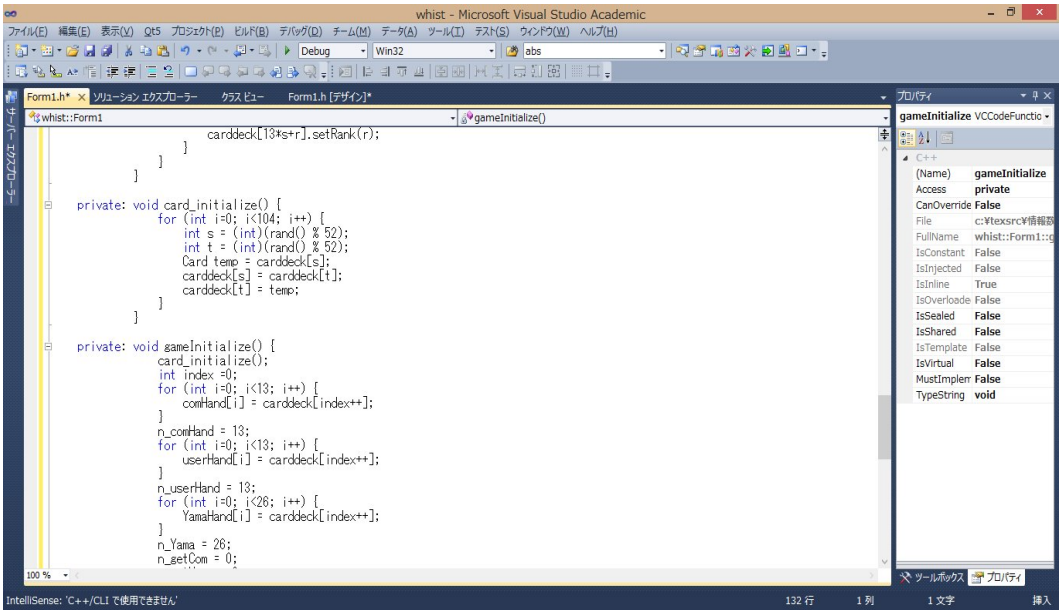
```
private: void card_initialize() {
    for (int i=0; i<104; i++) {
        int s = (int)(rand() % 52);
        int t = (int)(rand() % 52);
        Card temp = carddeck[s];
        carddeck[s] = carddeck[t];
        carddeck[t] = temp;
    }
}
```

です。これを

```
private: void gameInitialize() {
    card_initialize();
    int index =0;
    for (int i=0; i<13; i++) {
        comHand[i] = carddeck[index++];
```

```
    }
    n_comHand = 13;
    for (int i=0; i<13; i++) {
        userHand[i] = carddeck[index++];
    }
    n_userHand = 13;
    for (int i=0; i<26; i++) {
        YamaHand[i] = carddeck[index++];
    }
    n_Yama = 26;
    n_getCom = 0;
    n_getUser = 0;
    gameOverP = false;
    comPlayP = false;
    n_dispCom = 0;
}
```

の直前に打ち込みます。




```
    int index =0;
    for (int i=0; i<13; i++) {
        comHand[i] = carddeck[index++];
    }
    n_comHand = 13;
    for (int i=0; i<13; i++) {
        userHand[i] = carddeck[index++];
    }
    n_userHand = 13;
```

```
    for (int i=0; i<26; i++) {
        YamaHand[i] = carddeck[index++];
    }
    n_Yama = 26;
```

で、カードを配っています。

```
    n_getCom = 0;
    n_getUser = 0;
    gameOverP = false;
    comPlayP = false;
    n_dispCom = 0;
```

で、取ったカードがないこと、ゲームが終わっていないこと、コンピュータの手番でないこと、コンピュータがリードしていないことをセットしています。コンピュータの手札、ユーザーの手札、山の札、コンピュータが取ったカード、ユーザーが取ったカード、コンピュータがリードしたカードを保持する配列を定義しなければなりません。更に、ShowBan() の為に、WW と HH を定義します。

定義は

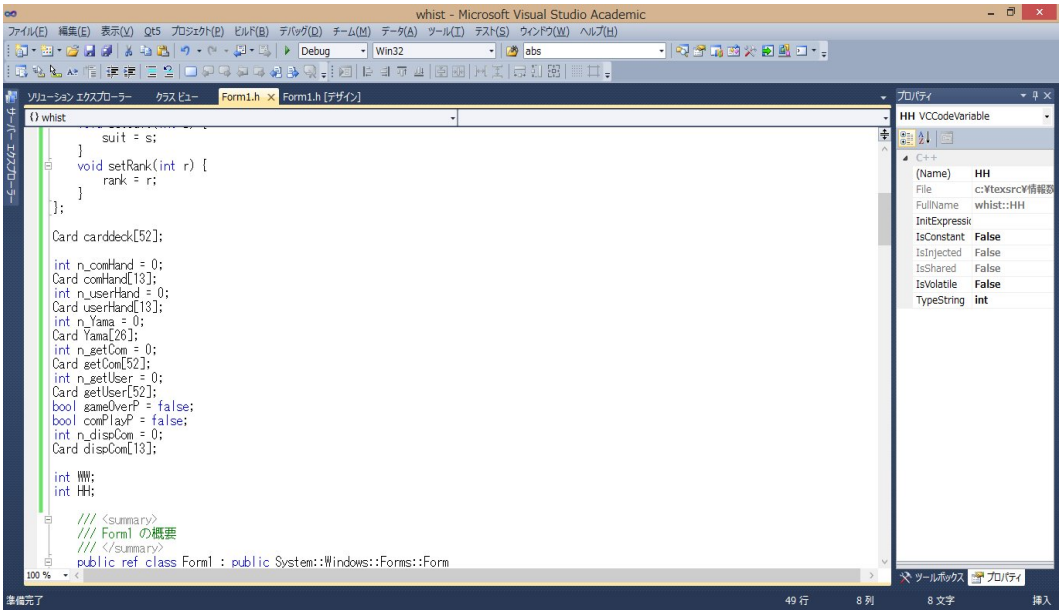
```
int n_comHand = 0;
Card comHand[13];
int n_userHand = 0;
Card userHand[13];
int n_Yama = 0;
Card Yama[26];
int n_getCom = 0;
Card getCom[52];
int n_getUser = 0;
Card getUser[52];
bool gameOverP = false;
bool comPlayP = false;
int n_dispCom = 0;
Card dispCom[13];

int WW;
int HH;
```

です。これを

```
struct Card {
   int suit;
   int rank;
   int getSuit() {
      return suit;
   }
```

```
    int getRank() {
        return rank;
    }
    void setSuit(int s) {
        suit = s;
    }
    void setRak(int r) {
        rank = r;
    }
};

Card carddeck[52];
```

の直後に打ち込みます。

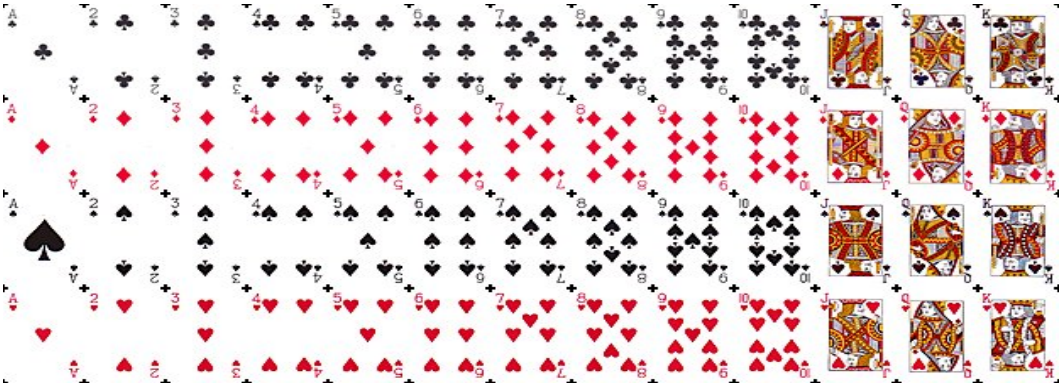


これでゲームを始める準備が出来ました。局面を表示します。

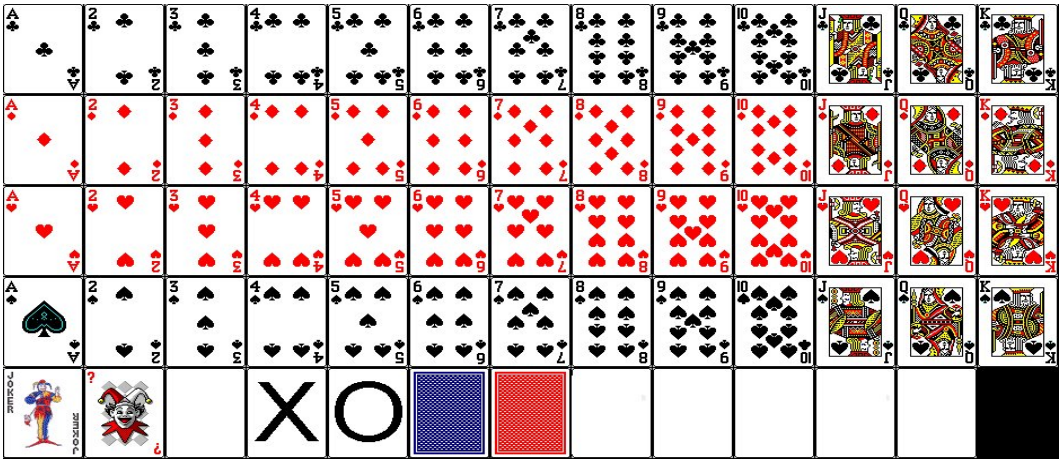
```
    ShowBan();
```

がそのための関数です。

カードを表示するためにはカードの画像が必要です。インターネットで探します。



や



のようなものを見つけ、前者を card256.gif 、後者を StandardDeck.jpg とします。

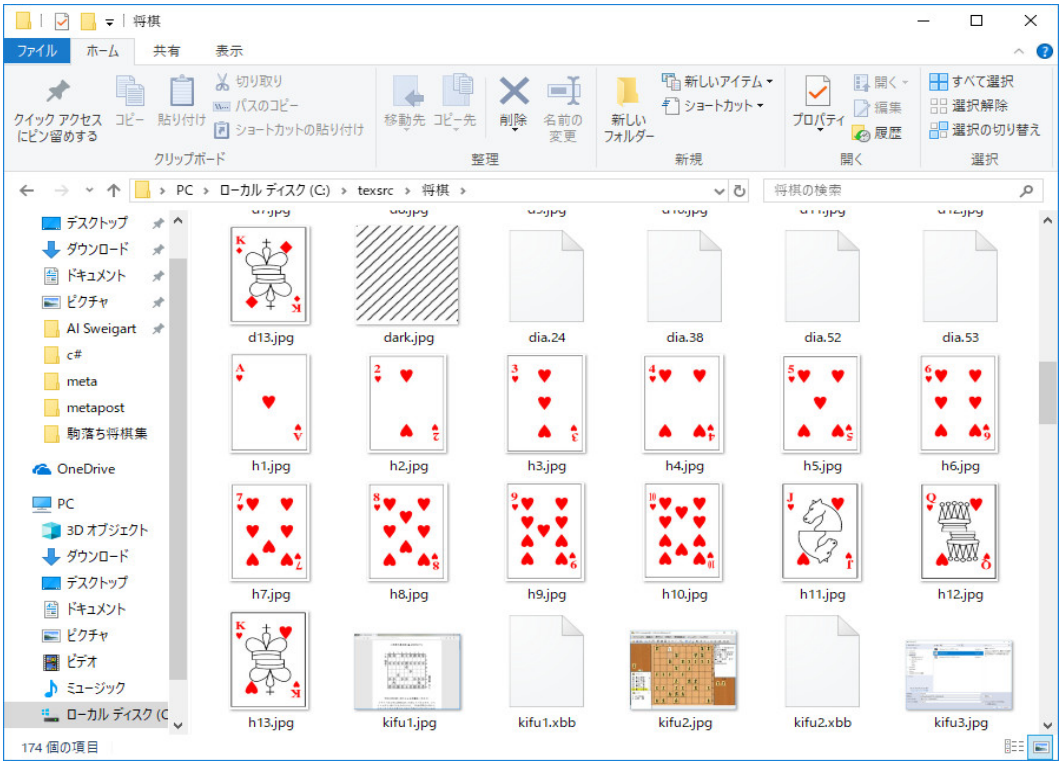
自分で楽しむだけでなければ、GIMP などで、例えば、スーツと数字だけの単純なオリジナルな画像を作れば良いです。

LaTeX と metapost と GIMP を使って、トランプの画像（.gif）を自分で作る方法を「情報数学の資料」のページの「Python による音声データの有効利用と LaTeX による将棋・囲碁カード作成補助の C#のソフト作成とカード画像作成」

http:/www.cc.kochi-u.ac.jp/~tyamag/jyohou/kifumodify.pdf

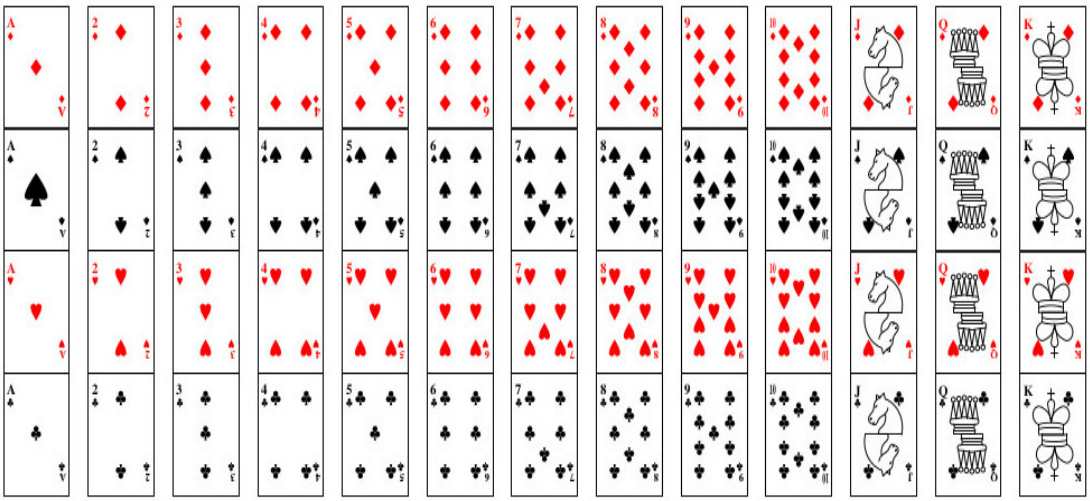
の pdf ファイルの最後に説明してあります。私は著作権を主張しませんから、それをそのまま使ってもいいですし、自分で好みの画像に改良して使ってもいいです。

私の作った metapost のプログラムでは次のような画像が作れます。



やっつけ仕事ですから、スーツのスペードやクラブやハートの形や大きさや絵札の絵が気に入らなければ、自分でプログラムを修正してください。

LaTeX を使って



のような画像を自分で作り、GIMP で加工すれば良いです。

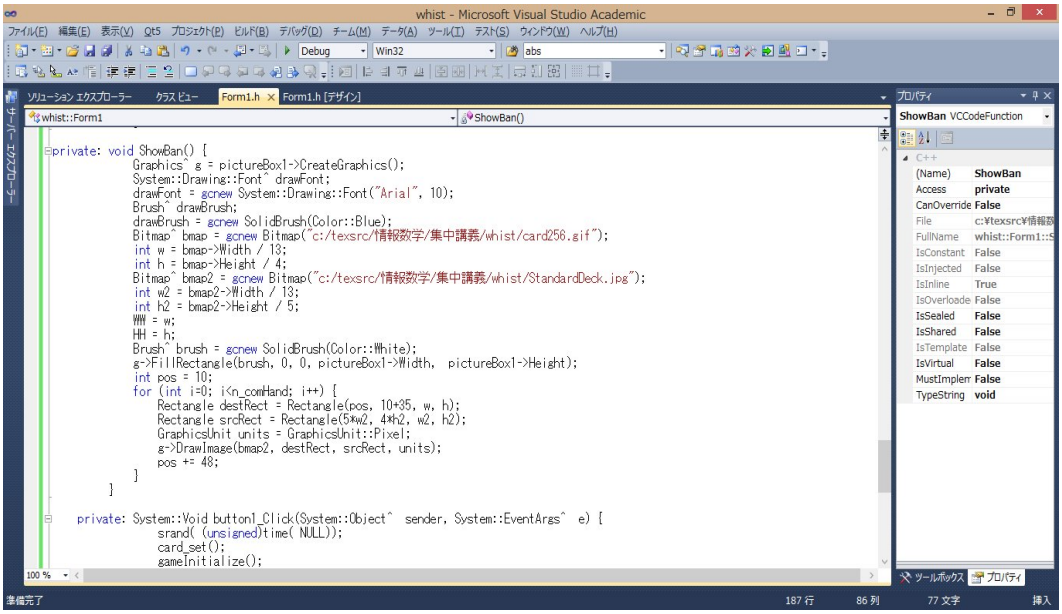
これらを使って、ひとまず

```
private: void ShowBan() {
   Graphics^ g = pictureBox1->CreateGraphics();
   System::Drawing::Font^ drawFont;
   drawFont = gcnew System::Drawing::Font("Arial", 10);
   Brush^ drawBrush;
   drawBrush = gcnew SolidBrush(Color::Blue);
   Bitmap^ bmap = gcnew Bitmap("c:/texsrc/情報数学/集中講義/whist/card256.gif");
   int w = bmap->Width / 13;
   int h = bmap->Height / 4;
   Bitmap^ bmap2 = gcnew Bitmap("c:/texsrc/情報数学/集中講義/whist/StandardDeck.jpg");
   int w2 = bmap2->Width / 13;
   int h2 = bmap2->Height / 5;
   WW = w;
   HH = h;
   Brush^ brush = gcnew SolidBrush(Color::White);
   g->FillRectangle(brush, 0, 0, pictureBox1->Width,  pictureBox1->Height);
   int pos = 10;
   for (int i=0; i<n_comHand; i++) {
      Rectangle destRect = Rectangle(pos, 10+35, w, h);
      Rectangle srcRect = Rectangle(5*w2, 4*h2, w2, h2);
      GraphicsUnit units = GraphicsUnit::Pixel;
      g->DrawImage(bmap2, destRect, srcRect, units);
      pos += 48;
   }
}
```

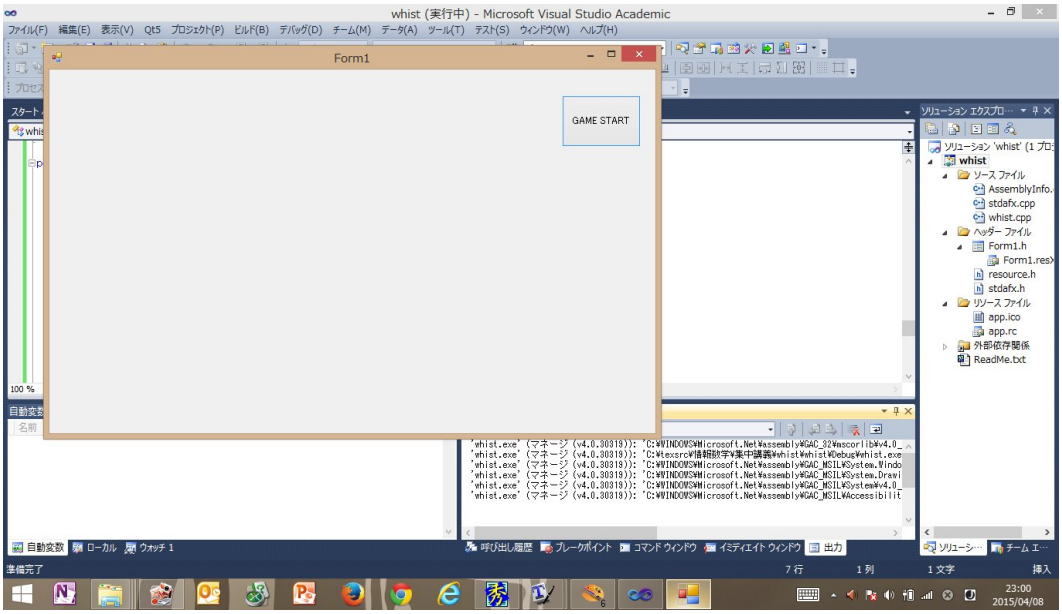
を

```
private: System::Void button1_Click(System::Object^  sender, System::EventArgs^  e) {
    srand( (unsigned)time( NULL));
    card_set();
    gameInitialize();
    ShowBan();
}
```

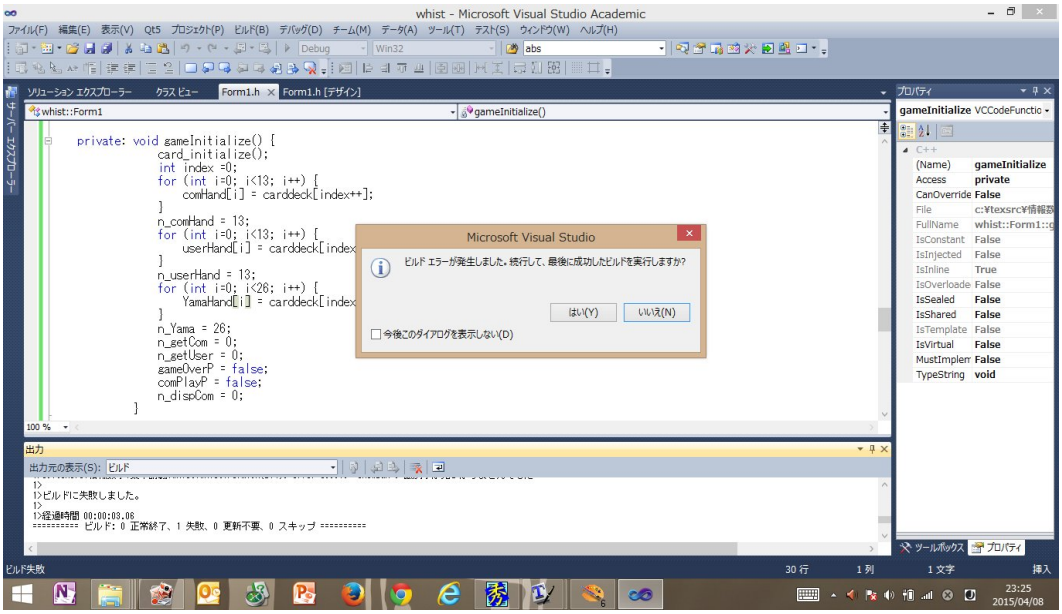
の直前に打ち込み



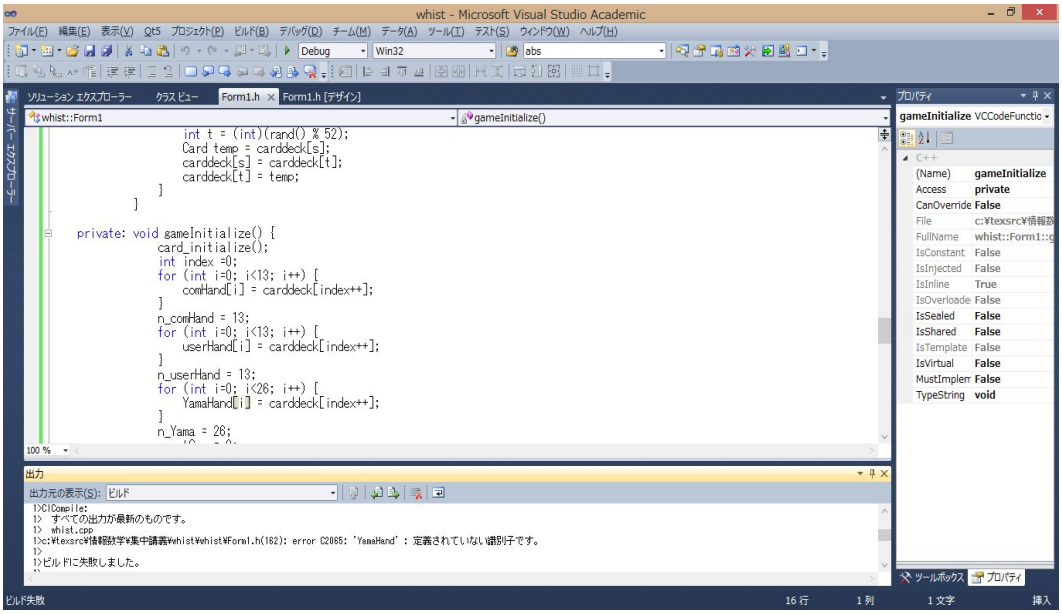
実行してみます。打ち間違えていなければ



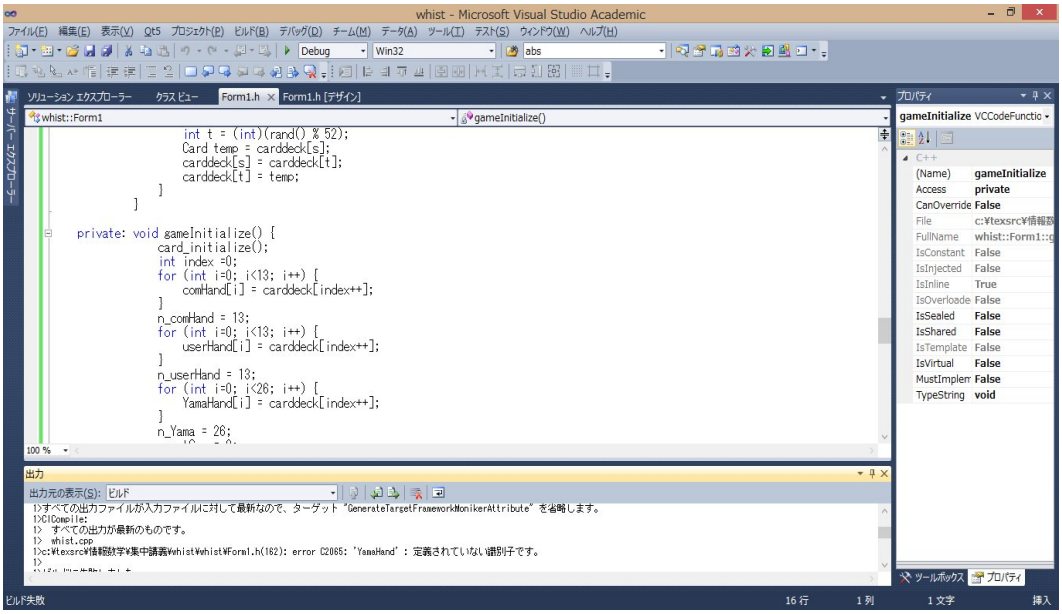
となります。ところが上のプログラムは間違えています。



となります。「いいえ」のボタンをクリックします。



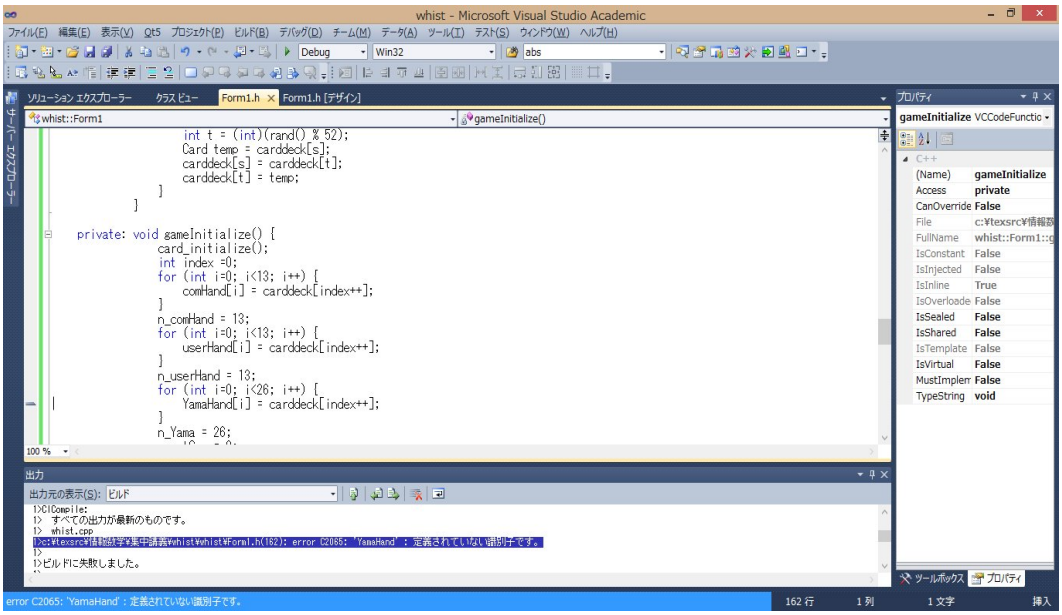
何処がエラーか調べます。



エラー表示の一番上に

'YamaHand' : 定義されていない識別子です。

と表示されています。この行をダブルクリックします。エラーのある行が表示されます。




```
private: void gameInitialize() {
    card_initialize();
    int index =0;
    for (int i=0; i<13; i++) {
        comHand[i] = carddeck[index++];
    }
```

```
    n_comHand = 13;
    for (int i=0; i<13; i++) {
        userHand[i] = carddeck[index++];
    }
    n_userHand = 13;
    for (int i=0; i<26; i++) {
        YamaHand[i] = carddeck[index++];
    }
    n_Yama = 26;
    n_getCom = 0;
    n_getUser = 0;
    gameOverP = false;
    comPlayP = false;
    n_dispCom = 0;
}
```

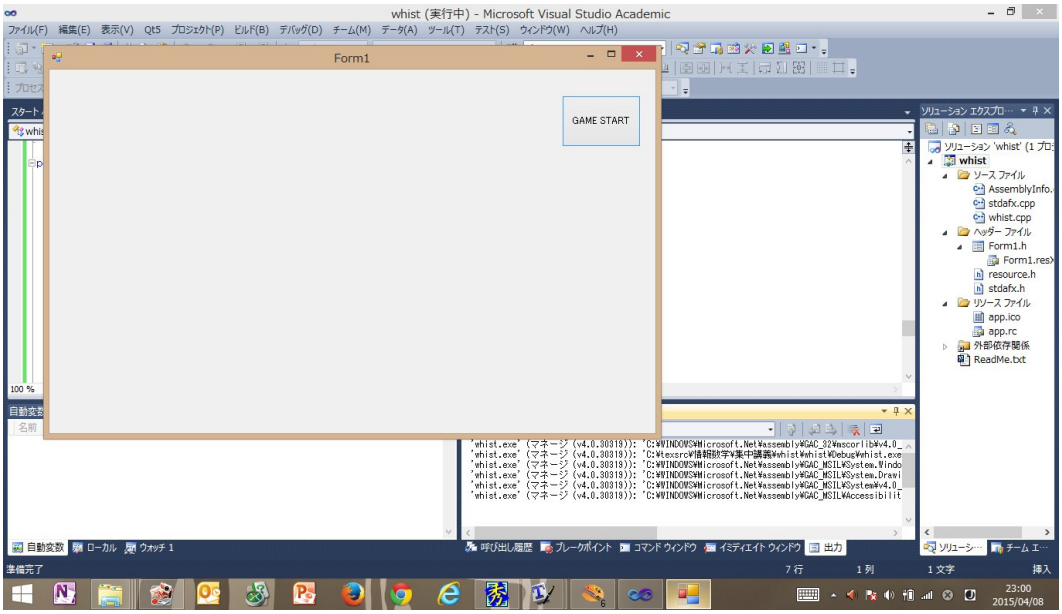
の

```
        YamaHand[i] = carddeck[index++];
```

を

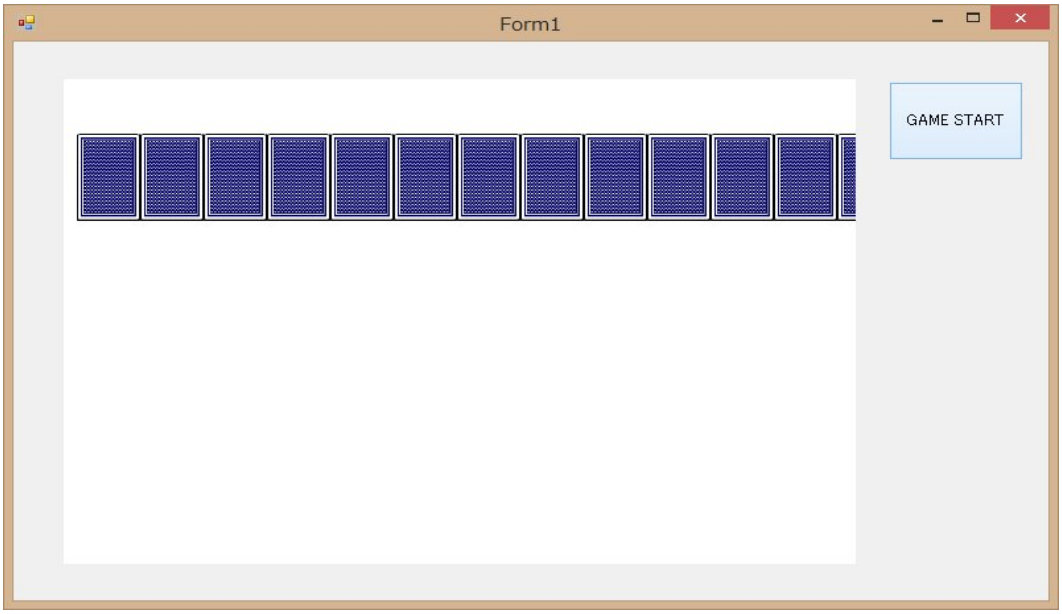
```
        Yama[i] = carddeck[index++];
```

と修正します。文法的な間違いはこのようにしてデバッグ出来ます。実行してみます。打ち間違えていなければ

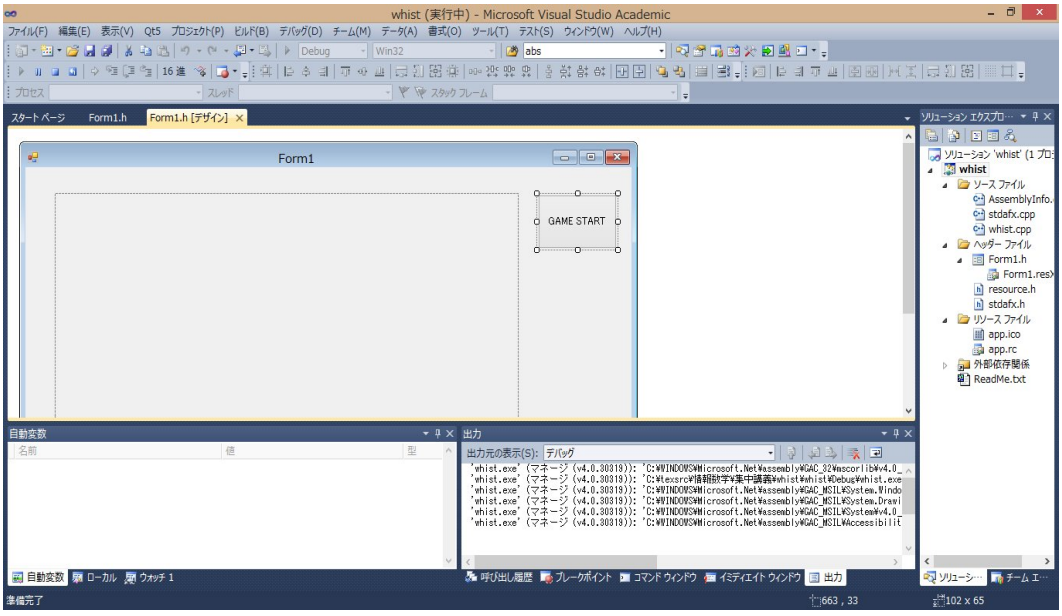


となります。

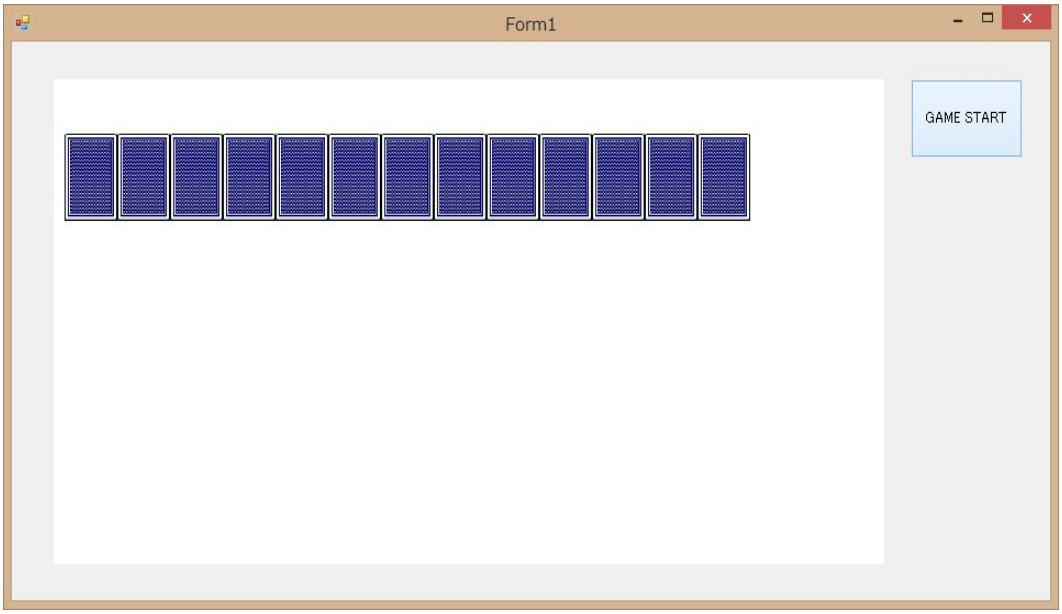
「GAME START」のボタンをクリックすれば



となります。表示領域が足りません。Form1.h「デザイン」をクリックして



Form を大きくし、ボタンを横に移動し、PictureBox を大きくします。



コンピュータの手札が 13 枚表示できるようになりました。
　全体のバランスを見ながら

```
    ShowBan();
```

をひとまず

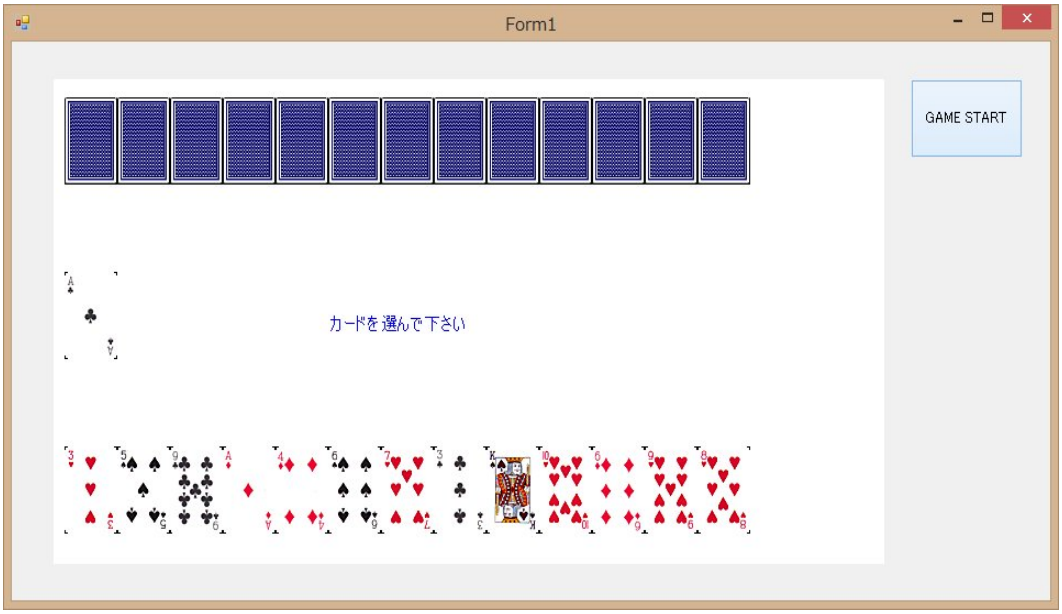
```
private: void ShowBan() {
   Graphics^ g = pictureBox1->CreateGraphics();
   System::Drawing::Font^ drawFont;
   drawFont = gcnew System::Drawing::Font("Arial", 10);
   Brush^ drawBrush;
   drawBrush = gcnew SolidBrush(Color::Blue);
   Bitmap^ bmap = gcnew Bitmap("c:/texsrc/情報数学/集中講義/whist/card256.gif");
   int w = bmap->Width / 13;
   int h = bmap->Height / 4;
   Bitmap^ bmap2 = gcnew Bitmap("c:/texsrc/情報数学/集中講義/whist/StandardDeck.jpg");
   int w2 = bmap2->Width / 13;
   int h2 = bmap2->Height / 5;
   WW = w;
   HH = h;
   Brush^ brush = gcnew SolidBrush(Color::White);
   g->FillRectangle(brush, 0, 0, pictureBox1->Width,  pictureBox1->Height);
   int pos = 10;
   for (int i=0; i<n_comHand; i++) {
      Rectangle destRect = Rectangle(pos, 10+5, w, h);
      Rectangle srcRect = Rectangle(5*w2, 4*h2, w2, h2);
      GraphicsUnit units = GraphicsUnit::Pixel;
```

```
   g->DrawImage(bmap2, destRect, srcRect, units);
   pos += 48;
}
pos += 100;
if (n_dispCom > 0) {
   int suit = dispCom[0].getSuit();
   int rank = dispCom[0].getRank();
   Rectangle destRect = Rectangle(pos, 10+5, w, h);
   Rectangle srcRect = Rectangle(rank*w, suit*h, w, h);
   GraphicsUnit units = GraphicsUnit::Pixel;
   g->DrawImage(bmap, destRect, srcRect, units);
}
pos = 10;
for (int i=0; i<n_getCom; i++) {
   int suit = getCom[i].getSuit();
   int rank = getCom[i].getRank();
   Rectangle destRect = Rectangle(pos, HH+10+5, w, h);
   Rectangle srcRect = Rectangle(rank*w, suit*h, w2, h2);
   GraphicsUnit units = GraphicsUnit::Pixel;
   g->DrawImage(bmap, destRect, srcRect, units);
   pos += 10;
}
pos = 10;
if (n_Yama > 0) {
   int suit = getCom[n_Yama-1].getSuit();
   int rank = getCom[n_Yama-1].getRank();
   Rectangle destRect = Rectangle(pos,2*HH+10+5, w, h);
   Rectangle srcRect = Rectangle(rank*w, suit*h, w, h);
   GraphicsUnit units = GraphicsUnit::Pixel;
   g->DrawImage(bmap, destRect, srcRect, units);
}
pos = 10;
for (int i=0; i<n_getUser; i++) {
   int suit = getUser[i].getSuit();
   int rank = getUser[i].getRank();
   Rectangle destRect = Rectangle(pos, 3*HH+10+5, w, h);
   Rectangle srcRect = Rectangle(rank*w, suit*h, w, h);
   GraphicsUnit units = GraphicsUnit::Pixel;
   g->DrawImage(bmap, destRect, srcRect, units);
   pos += 10;
}
pos = 10;
for (int i=0; i<n_userHand; i++) {
```

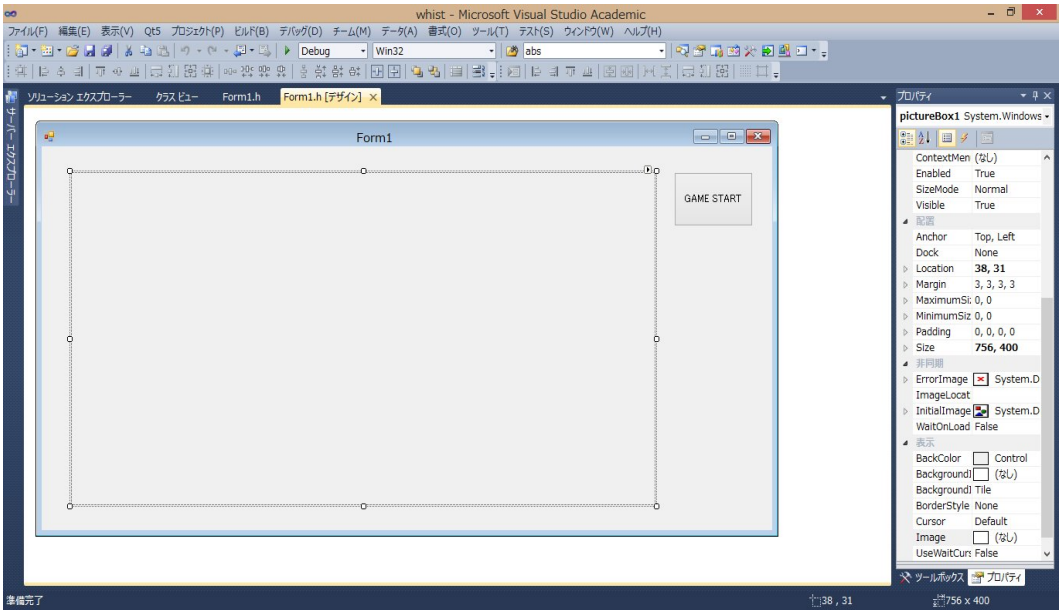
```
            int suit = userHand[i].getSuit();
            int rank = userHand[i].getRank();
            Rectangle destRect = Rectangle(pos, 4*HH+10+5, w, h);
            Rectangle srcRect = Rectangle(rank*w, suit*h, w, h);
            GraphicsUnit units = GraphicsUnit::Pixel;
            g->DrawImage(bmap, destRect, srcRect, units);
            pos += 48;
        }
        if (!gameOverP) {
            System::String^ drawString = "カードを選んで下さい";
            g->DrawString(drawString, drawFont, drawBrush, (float)250, (float)(2.5*HH+10+5));
        } else {
            int comValue = n_getCom / 2;
            int userValue = n_getUser / 2;
            System::String^drawString = "Game Over ";
            drawString += userValue + " : " + comValue;
            if (comValue > userValue) {
                drawString += " コンピュータの勝ちです。頑張ってね！";
                g->DrawString(drawString, drawFont, drawBrush, (float)250, (float)(2.5*HH+10+5));
            } else if (comValue < userValue) {
                drawString += " あなたの勝ちです。強いですね！";
                g->DrawString(drawString, drawFont, drawBrush, (float)250, (float)(2.5*HH+10+5));
            } else {
                drawString += "  引き分けです。もう一度しませんか？";
                g->DrawString(drawString, drawFont, drawBrush, (float)250, (float)(2.5*HH+10+5));
            }
        }
    }
```

とします。最初の部分も変えましたから注意してください。

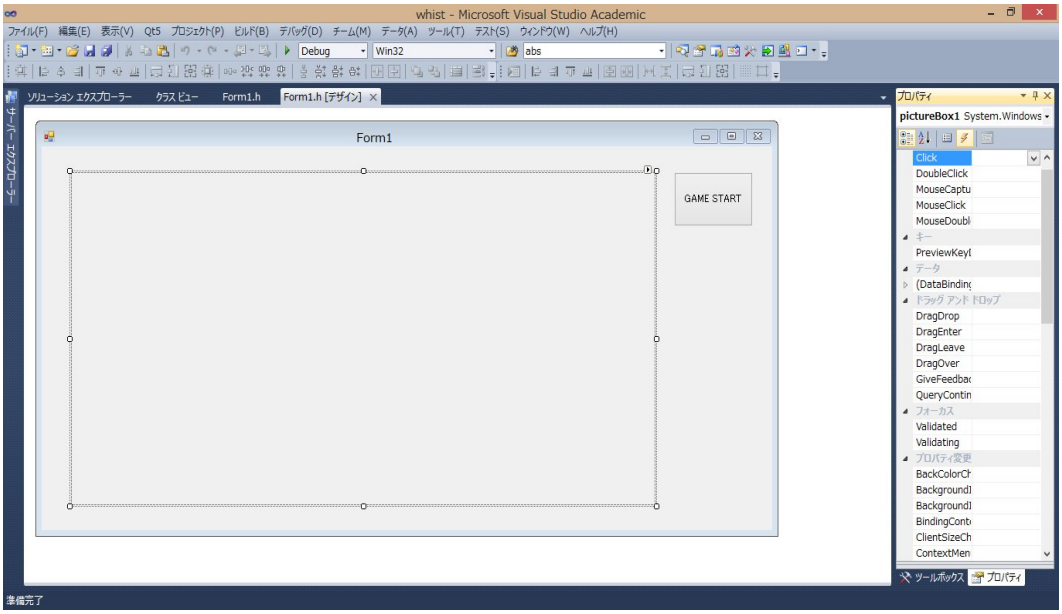
実行して、「GAME START」のボタンをクリックすれば



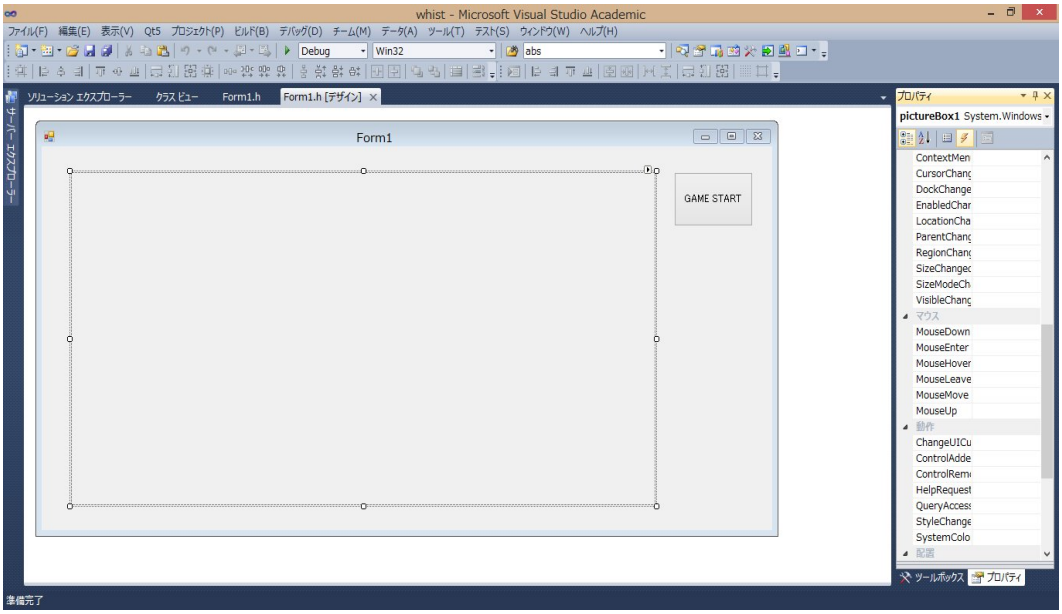
のような表示となります。実行するたびに表示が変わります。あまりよくシャッフルされていません。シャッフルのやり方がまずいみたいです、自分で工夫してください。カードを選んでリードする部分のプログラミングをします。これは PictureBox でマウスをクリックしたときに呼ばれる関数を使います。Form1.h「デザイン」で、 PictureBox をクリックし、



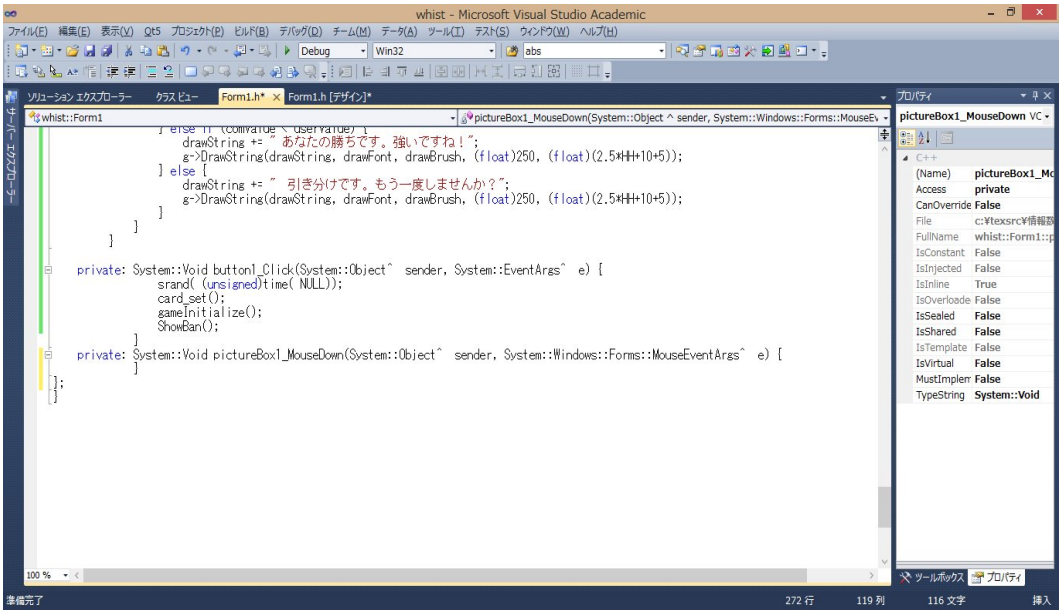
「プロパティ」で稲妻のようなボタンをクリックします。



中ほどの「マウス」の項目の「MouseDown」を



ダブルクリックします。




```
private: System::Void pictureBox1_MouseDown(System::Object^  sender,
     System::Windows::Forms::MouseEventArgs^  e) {
}
```

に処理を記述します。切り札のことを忘れていました。その前に

```
struct Card {
   int suit;
   int rank;
   int getSuit() {
      return suit;
   }
   int getRank() {
      return rank;
   }
   void setSuit(int s) {
      suit = s;
   }
   void setRank(int r) {
      rank = r;
   }
};

Card carddeck[52];

int n_comHand = 0;
Card comHand[13];
int n_userHand = 0;
```

```
Card userHand[13];
int n_Yama = 0;
Card Yama[26];
int n_getCom = 0;
Card getCom[52];
int n_getUser = 0;
Card getUser[52];
bool gameOverP = false;
bool comPlayP = false;
int n_dispCom = 0;
Card dispCom[13];

int WW;
int HH;
int trump;
```

のように、最後に

```
int trump;
```

を追加し、

```
private: void gameInitialize() {
   card_initialize();
   int index =0;
   for (int i=0; i<13; i++) {
      comHand[i] = carddeck[index++];
   }
   n_comHand = 13;
   for (int i=0; i<13; i++) {
      userHand[i] = carddeck[index++];
   }
   n_userHand = 13;
   for (int i=0; i<26; i++) {
      Yama[i] = carddeck[index++];
   }
   n_Yama = 26;
   n_getCom = 0;
   n_getUser = 0;
   gameOverP = false;
   comPlayP = false;
   n_dispCom = 0;
   trump = Yama[n_Yama-1].getSuit();
}
```

のように、最後に

```
   trump = Yama[n_Yama-1].getSuit();
```

を追加します。

```
private: System::Void pictureBox1_MouseDown(System::Object^  sender,
     System::Windows::Forms::MouseEventArgs^  e) {
}
```

を

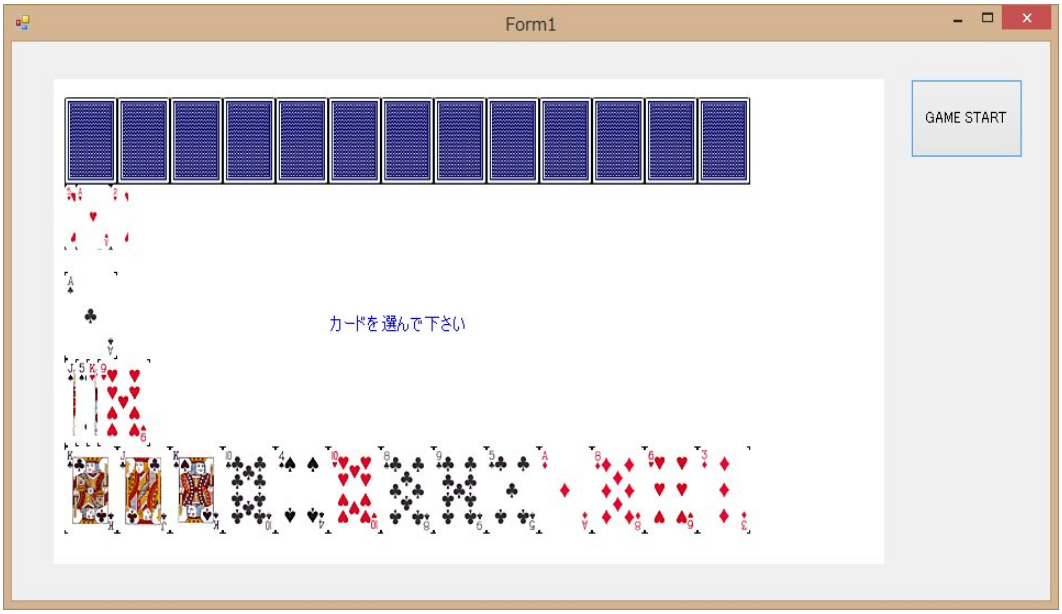
```
private: System::Void pictureBox1_MouseDown(System::Object^  sender,
     System::Windows::Forms::MouseEventArgs^  e) {
   int mX = e->X;
   int mY = e->Y;
   int userIndex = -1;
   int pos = 10;
   for (int i=0; i<n_userHand; i++) {
      if ((mX>pos && mX<pos+WW)&& (mY>4*HH+10+5 && mY<4*HH+82+5)){
         userIndex = i;
         break;
      }
      pos += WW;
   }
   if (userIndex < 0)
      return;
   if (!comPlayP) {
      int suit = userHand[userIndex].suit;
      int rank = userHand[userIndex].rank;
      int comIndex = -1;
      for (int i=0; i<n_comHand; i++) {
         if (comHand[i].suit == suit) {
            comIndex = i;
            break;
         }
      }
      if (comIndex < 0) {
         for (int i=0; i<n_comHand; i++) {
            if (comHand[i].suit == trump) {
               comIndex = i;
               break;
            }
         }
         if (comIndex < 0) {
```

```
            comIndex = (int)(rand() % n_comHand);
        }
    }
    if (comHand[comIndex].suit == userHand[userIndex].suit) {
        if (comHand[comIndex].rank == 0) {
            getCom[n_getCom++] = userHand[userIndex];
            getCom[n_getCom++] = comHand[comIndex];
            comPlayP = true;
        } else if (userHand[userIndex].rank == 0) {
            getUser[n_getUser++] = userHand[userIndex];
            getUser[n_getUser++] = comHand[comIndex];
            comPlayP = false;
        } else if (comHand[comIndex].rank > userHand[userIndex].rank) {
            getCom[n_getCom++] = userHand[userIndex];
            getCom[n_getCom++] = comHand[comIndex];
            comPlayP = true;
        } else {
            getUser[n_getUser++] = userHand[userIndex];
            getUser[n_getUser++] = comHand[comIndex];
            comPlayP = false;
        }
    } else if (comHand[comIndex].suit == trump) {
        getCom[n_getCom++] = userHand[userIndex];
        getCom[n_getCom++] = comHand[comIndex];
        comPlayP = true;
    } else if (userHand[userIndex].suit == trump) {
        getUser[n_getUser++] = userHand[userIndex];
        getUser[n_getUser++] = comHand[comIndex];
        comPlayP = false;
    } else {
        getUser[n_getUser++] = userHand[userIndex];
        getUser[n_getUser++] = comHand[comIndex];
        comPlayP = false;
    }
    for (int i=userIndex; i<n_userHand-1; i++)
        userHand[i] = userHand[i+1];
    n_userHand--;
    for (int i=comIndex; i<n_comHand-1; i++)
        comHand[i] = comHand[i+1];
    n_comHand--;
} else {
    int suit = userHand[userIndex].suit;
    int rank = userHand[userIndex].rank;
```

```
if (suit != dispCom[0].suit) {
   bool flag = false;
   for (int i=0; i<n_userHand; i++) {
      if (userHand[i].suit == dispCom[0].suit) {
         flag = true; break;
      }
   }
   if (flag) {
      System::String^ str ="リードは";
      switch (dispCom[0].suit) {
         case 0: str += "クラブ"; break;
         case 1: str += "ダイヤ"; break;
         case 2: str += "スペード"; break;
         case 3: str += "ハート"; break;
      }
      str += "です";
      MessageBox::Show(str);
      return;
   }
}
if (dispCom[0].suit == userHand[userIndex].suit) {
   if (dispCom[0].rank == 0) {
      getCom[n_getCom++] = userHand[userIndex];
      getCom[n_getCom++] = dispCom[0];
      comPlayP = true;
   } else if (userHand[userIndex].rank == 0) {
      getUser[n_getUser++] = userHand[userIndex];
      getUser[n_getUser++] = dispCom[0];
      comPlayP = false;
   } else if (dispCom[0].rank > userHand[userIndex].rank) {
      getCom[n_getCom++] = userHand[userIndex];
      getCom[n_getCom++] = dispCom[0];
      comPlayP = true;
   } else {
      getUser[n_getUser++] = userHand[userIndex];
      getUser[n_getUser++] = dispCom[0];
      comPlayP = false;
   }
} else if (dispCom[0].suit == trump) {
   getCom[n_getCom++] = userHand[userIndex];
   getCom[n_getCom++] = dispCom[0];
   comPlayP = true;
} else if (userHand[userIndex].suit == trump) {
```

```
            getUser[n_getUser++] = userHand[userIndex];
            getUser[n_getUser++] = dispCom[0];
            comPlayP = false;
        } else {
            getCom[n_getCom++] = userHand[userIndex];
            getCom[n_getCom++] = dispCom[0];
            comPlayP = true;
        }
        for (int i=userIndex; i<n_userHand-1; i++)
            userHand[i] = userHand[i+1];
            n_userHand--;
        }
        if (n_Yama > 0) {
            if (comPlayP) {
                comHand[n_comHand++] = Yama[n_Yama-1];
                n_Yama--;
                userHand[n_userHand++] = Yama[n_Yama-1];
                n_Yama--;
            } else {
                userHand[n_comHand++] = Yama[n_Yama-1];
                n_Yama--;
                comHand[n_userHand++] = Yama[n_Yama-1];
                n_Yama--;
            }
        }
        if (n_comHand == 0) {
            gameOverP = true;
            n_dispCom = 0;
        } else if (comPlayP) {
            int comIndex = (int)(rand() % n_comHand);
            dispCom[0] = comHand[comIndex];
            n_dispCom = 1;
            for (int i=comIndex; i<n_comHand-1; i++)
                comHand[i] = comHand[i+1];
            n_comHand--;
        } else {
            n_dispCom = 0;
        }
        ShowBan();
    }
```

とします。多分、プログラムが完成したと思って、実行してみます。



表示がおかしいです。ShowBan() が間違っています。「コピー・ペースト」でプログラムを作るとこんな間違いはしょっちゅうです。

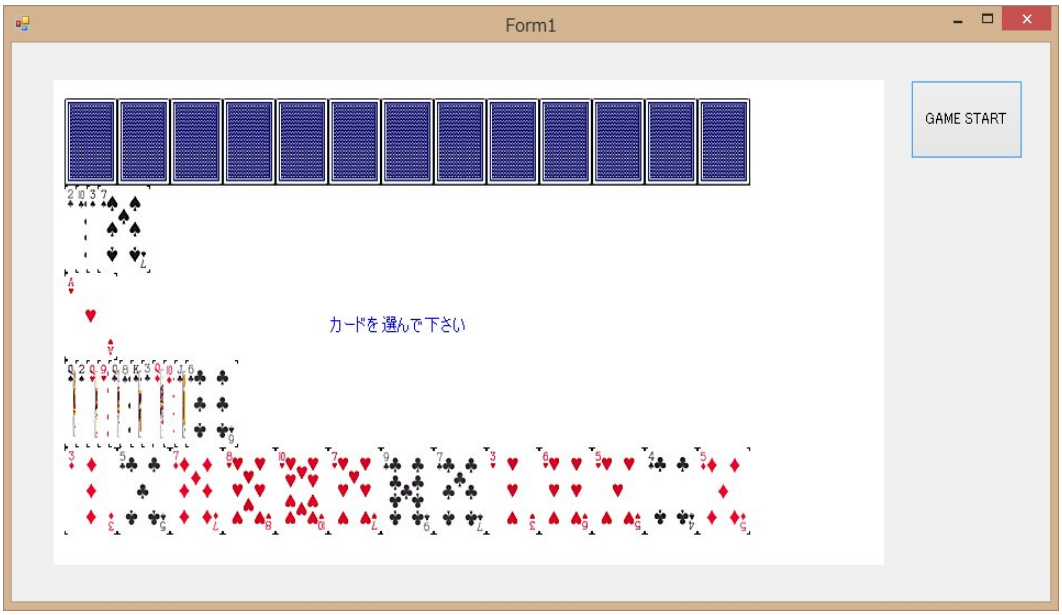
```
pos = 10;
for (int i=0; i<n_getCom; i++) {
int suit = getCom[i].getSuit();
int rank = getCom[i].getRank();
Rectangle destRect = Rectangle(pos, HH+10+5, w, h);
Rectangle srcRect = Rectangle(rank*w, suit*h, w2, h2);
GraphicsUnit units = GraphicsUnit::Pixel;
g->DrawImage(bmap, destRect, srcRect, units);
pos += 10;
}
pos = 10;
if (n_Yama > 0) {
int suit = getCom[n_Yama-1].getSuit();
int rank = getCom[n_Yama-1].getRank();
Rectangle destRect = Rectangle(pos,2*HH+10+5, w, h);
Rectangle srcRect = Rectangle(rank*w, suit*h, w, h);
GraphicsUnit units = GraphicsUnit::Pixel;
g->DrawImage(bmap, destRect, srcRect, units);
}
```

の部分を

```
pos = 10;
for (int i=0; i<n_getCom; i++) {
int suit = getCom[i].getSuit();
```

```
int rank = getCom[i].getRank();
Rectangle destRect = Rectangle(pos, HH+10+5, w, h);
Rectangle srcRect = Rectangle(rank*w, suit*h, w, h);
GraphicsUnit units = GraphicsUnit::Pixel;
g->DrawImage(bmap, destRect, srcRect, units);
pos += 10;
}
pos = 10;
if (n_Yama > 0) {
int suit = Yama[n_Yama-1].getSuit();
int rank = Yama[n_Yama-1].getRank();
Rectangle destRect = Rectangle(pos,2*HH+10+5, w, h);
Rectangle srcRect = Rectangle(rank*w, suit*h, w, h);
GraphicsUnit units = GraphicsUnit::Pixel;
g->DrawImage(bmap, destRect, srcRect, units);
}
```

と直します。実行してみます。



正しく動いているみたいです。

最終的な Form1.h は

```
#pragma once
#include <stdlib.h>
#include <time.h>

namespace whist {
   using namespace System;
   using namespace System::ComponentModel;
```

```
   using namespace System::Collections;
   using namespace System::Windows::Forms;
   using namespace System::Data;
   using namespace System::Drawing;

struct Card {
   int suit;
   int rank;
   int getSuit() {
      return suit;
   }
   int getRank() {
      return rank;
   }
   void setSuit(int s) {
      suit = s;
   }
   void setRank(int r) {
     rank = r;
   }
};

Card carddeck[52];

int n_comHand = 0;
Card comHand[13];
int n_userHand = 0;
Card userHand[13];
int n_Yama = 0;
Card Yama[26];
int n_getCom = 0;
Card getCom[52];
int n_getUser = 0;
Card getUser[52];
bool gameOverP = false;
bool comPlayP = false;
int n_dispCom = 0;
Card dispCom[13];

int WW;
int HH;
int trump;
```

```
   /// <summary>
   /// Form1 の概要
   /// </summary>
   public ref class Form1 : public System::Windows::Forms::Form
{
   public:
      Form1(void)
      {
         InitializeComponent();
         //
         //TODO: ここにコンストラクター コードを追加します
         //
     }

     protected:
     /// <summary>
     /// 使用中のリソースをすべてクリーンアップします。
         /// </summary>
         ~Form1()
         {
             if (components)
             {
                 delete components;
             }
         }
     private: System::Windows::Forms::PictureBox^  pictureBox1;
     private: System::Windows::Forms::Button^  button1;
     protected:

     private:
         /// <summary>
         /// 必要なデザイナー変数です。
         /// </summary>
         System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code
         /// <summary>
         /// デザイナー サポートに必要なメソッドです。このメソッドの内容を
         /// コード エディターで変更しないでください。
         /// </summary>
         void InitializeComponent(void)
         {
             this->pictureBox1 = (gcnew System::Windows::Forms::PictureBox());
```

```
            this->button1 = (gcnew System::Windows::Forms::Button());
            (cli::safe_cast<System::ComponentModel::ISupportInitialize^  >
                (this->pictureBox1))->BeginInit();
            this->SuspendLayout();
            //
            // pictureBox1
            //
            this->pictureBox1->Location = System::Drawing::Point(38, 31);
            this->pictureBox1->Name = L"pictureBox1";
            this->pictureBox1->Size = System::Drawing::Size(756, 400);
            this->pictureBox1->TabIndex = 0;
            this->pictureBox1->TabStop = false;
            this->pictureBox1->MouseDown += gcnew
 System::Windows::Forms::MouseEventHandler(this, &Form1::pictureBox1_MouseDown);
            //
            // button1
            //
            this->button1->Location = System::Drawing::Point(818, 31);
            this->button1->Name = L"button1";
            this->button1->Size = System::Drawing::Size(102, 65);
            this->button1->TabIndex = 1;
            this->button1->Text = L"GAME START";
            this->button1->UseVisualStyleBackColor = true;
            this->button1->Click += gcnew System::EventHandler(this, &Form1::button1_Click);
            //
            // Form1
            //
            this->AutoScaleDimensions = System::Drawing::SizeF(6, 12);
            this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;
            this->ClientSize = System::Drawing::Size(945, 461);
            this->Controls->Add(this->button1);
            this->Controls->Add(this->pictureBox1);
            this->Name = L"Form1";
            this->Text = L"Form1";
            (cli::safe_cast<System::ComponentModel::ISupportInitialize^  >
                    (this->pictureBox1))->EndInit();
            this->ResumeLayout(false);

        }
#pragma endregion
private: void card_set() {
     for (int s=0; s<4; s++) {
         for (int r=0; r<13; r++) {
```

```
            carddeck[13*s+r].setSuit(s);
            carddeck[13*s+r].setRank(r);
        }
    }
}

private: void card_initialize() {
    for (int i=0; i<104; i++) {
        int s = (int)(rand() % 52);
        int t = (int)(rand() % 52);
        Card temp = carddeck[s];
        carddeck[s] = carddeck[t];
        carddeck[t] = temp;
    }
}

private: void gameInitialize() {
    card_initialize();
    int index =0;
    for (int i=0; i<13; i++) {
        comHand[i] = carddeck[index++];
    }
    n_comHand = 13;
    for (int i=0; i<13; i++) {
        userHand[i] = carddeck[index++];
    }
    n_userHand = 13;
    for (int i=0; i<26; i++) {
        Yama[i] = carddeck[index++];
    }
    n_Yama = 26;
    n_getCom = 0;
    n_getUser = 0;
    gameOverP = false;
    comPlayP = false;
    n_dispCom = 0;
    trump = Yama[n_Yama-1].getSuit();
 }

private: void ShowBan() {
    Graphics^ g = pictureBox1->CreateGraphics();
    System::Drawing::Font^ drawFont;
    drawFont = gcnew System::Drawing::Font("Arial", 10);
```

```
Brush^ drawBrush;
drawBrush = gcnew SolidBrush(Color::Blue);
Bitmap^ bmap = gcnew Bitmap("c:/texsrc/情報数学/集中講義/whist/card256.gif");
int w = bmap->Width / 13;
int h = bmap->Height / 4;
Bitmap^ bmap2 = gcnew Bitmap("c:/texsrc/情報数学/集中講義/whist/StandardDeck.jpg");
int w2 = bmap2->Width / 13;
int h2 = bmap2->Height / 5;
WW = w;
HH = h;
Brush^ brush = gcnew SolidBrush(Color::White);
g->FillRectangle(brush, 0, 0, pictureBox1->Width,  pictureBox1->Height);
int pos = 10;
for (int i=0; i<n_comHand; i++) {
    Rectangle destRect = Rectangle(pos, 10+5, w, h);
    Rectangle srcRect = Rectangle(5*w2, 4*h2, w2, h2);
    GraphicsUnit units = GraphicsUnit::Pixel;
    g->DrawImage(bmap2, destRect, srcRect, units);
    pos += 48;
}
pos += 100;
if (n_dispCom > 0) {
    int suit = dispCom[0].getSuit();
    int rank = dispCom[0].getRank();
    Rectangle destRect = Rectangle(pos, 10+5, w, h);
    Rectangle srcRect = Rectangle(rank*w, suit*h, w, h);
    GraphicsUnit units = GraphicsUnit::Pixel;
    g->DrawImage(bmap, destRect, srcRect, units);
}
pos = 10;
for (int i=0; i<n_getCom; i++) {
    int suit = getCom[i].getSuit();
    int rank = getCom[i].getRank();
    Rectangle destRect = Rectangle(pos, HH+10+5, w, h);
    Rectangle srcRect = Rectangle(rank*w, suit*h, w, h);
    GraphicsUnit units = GraphicsUnit::Pixel;
    g->DrawImage(bmap, destRect, srcRect, units);
    pos += 10;
}
pos = 10;
if (n_Yama > 0) {
    int suit = Yama[n_Yama-1].getSuit();
    int rank = Yama[n_Yama-1].getRank();
```

```
    Rectangle destRect = Rectangle(pos,2*HH+10+5, w, h);
    Rectangle srcRect = Rectangle(rank*w, suit*h, w, h);
    GraphicsUnit units = GraphicsUnit::Pixel;
    g->DrawImage(bmap, destRect, srcRect, units);
}
pos = 10;
for (int i=0; i<n_getUser; i++) {
    int suit = getUser[i].getSuit();
    int rank = getUser[i].getRank();
    Rectangle destRect = Rectangle(pos, 3*HH+10+5, w, h);
    Rectangle srcRect = Rectangle(rank*w, suit*h, w, h);
    GraphicsUnit units = GraphicsUnit::Pixel;
    g->DrawImage(bmap, destRect, srcRect, units);
    pos += 10;
}
pos = 10;
for (int i=0; i<n_userHand; i++) {
    int suit = userHand[i].getSuit();
    int rank = userHand[i].getRank();
    Rectangle destRect = Rectangle(pos, 4*HH+10+5, w, h);
    Rectangle srcRect = Rectangle(rank*w, suit*h, w, h);
    GraphicsUnit units = GraphicsUnit::Pixel;
    g->DrawImage(bmap, destRect, srcRect, units);
    pos += 48;
}
if (!gameOverP) {
    System::String^ drawString = "カードを選んで下さい";
    g->DrawString(drawString, drawFont, drawBrush, (float)250, (float)(2.5*HH+10+5));
} else {
    int comValue = n_getCom / 2;
    int userValue = n_getUser / 2;
    System::String^drawString = "Game Over ";
    drawString += userValue + " : " + comValue;
    if (comValue > userValue) {
        drawString += " コンピュータの勝ちです。頑張ってね！";
        g->DrawString(drawString, drawFont, drawBrush, (float)250, (float)(2.5*HH+10+5));
    } else if (comValue < userValue) {
        drawString += " あなたの勝ちです。強いですね！";
        g->DrawString(drawString, drawFont, drawBrush, (float)250, (float)(2.5*HH+10+5));
    } else {
        drawString += "  引き分けです。もう一度しませんか？";
        g->DrawString(drawString, drawFont, drawBrush, (float)250, (float)(2.5*HH+10+5));
    }
```

```
    }
}

private: System::Void button1_Click(System::Object^  sender, System::EventArgs^  e) {
    srand( (unsigned)time( NULL));
    card_set();
    gameInitialize();
    ShowBan();
}
private: System::Void pictureBox1_MouseDown(System::Object^  sender,
         System::Windows::Forms::MouseEventArgs^  e) {
    int mX = e->X;
    int mY = e->Y;
    int userIndex = -1;
    int pos = 10;
    for (int i=0; i<n_userHand; i++) {
        if ((mX>pos && mX<pos+WW)&& (mY>4*HH+10+5 && mY<4*HH+82+5)){
            userIndex = i;
            break;
        }
        pos += WW;
    }
    if (userIndex < 0)
        return;
    if (!comPlayP) {
        int suit = userHand[userIndex].suit;
        int rank = userHand[userIndex].rank;
        int comIndex = -1;
        int cand[13];
        int n_cand = 0;
        for (int i=0; i<n_comHand; i++) {
            if (comHand[i].suit == suit) {
                comIndex = i;
                break;
            }
        }
        if (comIndex < 0) {
            for (int i=0; i<n_comHand; i++) {
                if (comHand[i].suit == trump) {
                    comIndex = i;
                    break;
                }
            }
```

```
            if (comIndex < 0) {
                comIndex = (int)(rand() % n_comHand);
            }
        }
        if (comHand[comIndex].suit == userHand[userIndex].suit) {
            if (comHand[comIndex].rank == 0) {
                getCom[n_getCom++] = userHand[userIndex];
                getCom[n_getCom++] = comHand[comIndex];
                comPlayP = true;
            } else if (userHand[userIndex].rank == 0) {
                getUser[n_getUser++] = userHand[userIndex];
                getUser[n_getUser++] = comHand[comIndex];
                comPlayP = false;
            } else if (comHand[comIndex].rank > userHand[userIndex].rank) {
                getCom[n_getCom++] = userHand[userIndex];
                getCom[n_getCom++] = comHand[comIndex];
                comPlayP = true;
            } else {
                getUser[n_getUser++] = userHand[userIndex];
                getUser[n_getUser++] = comHand[comIndex];
                comPlayP = false;
            }
        } else if (comHand[comIndex].suit == trump) {
            getCom[n_getCom++] = userHand[userIndex];
            getCom[n_getCom++] = comHand[comIndex];
            comPlayP = true;
        } else if (userHand[userIndex].suit == trump) {
            getUser[n_getUser++] = userHand[userIndex];
            getUser[n_getUser++] = comHand[comIndex];
            comPlayP = false;
        } else {
            getUser[n_getUser++] = userHand[userIndex];
            getUser[n_getUser++] = comHand[comIndex];
            comPlayP = false;
        }
        for (int i=userIndex; i<n_userHand-1; i++)
            userHand[i] = userHand[i+1];
        n_userHand--;
        for (int i=comIndex; i<n_comHand-1; i++)
            comHand[i] = comHand[i+1];
        n_comHand--;
    } else {
        int suit = userHand[userIndex].suit;
```

```
int rank = userHand[userIndex].rank;
if (suit != dispCom[0].suit) {
    bool flag = false;
    for (int i=0; i<n_userHand; i++) {
        if (userHand[i].suit == dispCom[0].suit) {
            flag = true; break;
        }
    }
    if (flag) {
        System::String^ str ="リードは";
        switch (dispCom[0].suit) {
            case 0: str += "クラブ"; break;
            case 1: str += "ダイヤ"; break;
            case 2: str += "スペード"; break;
            case 3: str += "ハート"; break;
        }
        str += "です";
        MessageBox::Show(str);
        return;
    }
}
if (dispCom[0].suit == userHand[userIndex].suit) {
    if (dispCom[0].rank == 0) {
        getCom[n_getCom++] = userHand[userIndex];
        getCom[n_getCom++] = dispCom[0];
        comPlayP = true;
    } else if (userHand[userIndex].rank == 0) {
        getUser[n_getUser++] = userHand[userIndex];
        getUser[n_getUser++] = dispCom[0];
        comPlayP = false;
    } else if (dispCom[0].rank > userHand[userIndex].rank) {
        getCom[n_getCom++] = userHand[userIndex];
        getCom[n_getCom++] = dispCom[0];
        comPlayP = true;
    } else {
        getUser[n_getUser++] = userHand[userIndex];
        getUser[n_getUser++] = dispCom[0];
        comPlayP = false;
    }
} else if (dispCom[0].suit == trump) {
    getCom[n_getCom++] = userHand[userIndex];
    getCom[n_getCom++] = dispCom[0];
    comPlayP = true;
```

```
        } else if (userHand[userIndex].suit == trump) {
            getUser[n_getUser++] = userHand[userIndex];
            getUser[n_getUser++] = dispCom[0];
            comPlayP = false;
        } else {
            getCom[n_getCom++] = userHand[userIndex];
            getCom[n_getCom++] = dispCom[0];
            comPlayP = true;
        }
        for (int i=userIndex; i<n_userHand-1; i++)
            userHand[i] = userHand[i+1];
            n_userHand--;
        }
        if (n_Yama > 0) {
            if (comPlayP) {
                comHand[n_comHand++] = Yama[n_Yama-1];
                n_Yama--;
                userHand[n_userHand++] = Yama[n_Yama-1];
                n_Yama--;
            } else {
                userHand[n_comHand++] = Yama[n_Yama-1];
                n_Yama--;
                comHand[n_userHand++] = Yama[n_Yama-1];
                n_Yama--;
            }
        }
        if (n_comHand == 0) {
            gameOverP = true;
            n_dispCom = 0;
        } else if (comPlayP) {
            int comIndex = (int)(rand() % n_comHand);
            dispCom[0] = comHand[comIndex];
            n_dispCom = 1;
            for (int i=comIndex; i<n_comHand-1; i++)
                comHand[i] = comHand[i+1];
            n_comHand--;
        } else {
            n_dispCom = 0;
        }
        ShowBan();
    }
};
}
```

です。

ユーザーの手札を綺麗にソートした方が見やすいです。コンピュータの思考は単にルール通り手を選んでいるだけで、強くなる工夫はしていません。自分で考えて下さい。コンピュータの手札もソートすればコンピュータがどのカードを出すべきか計算するのに都合がいいです。更に、ユーザーが取った表示されている山のカードを覚えておく配列も作るべきです。それらの情報を元に戦略を考えるべきです。このプログラムは未完成です。どうすれば強くなるかを考えるのは演習とします。また、普通の文章と違って、私がプログラムを作るときのプロセスをかなり忠実に再現したので、つまり、プログラムが完成してから文章を作ったのではなく、プログラムを作りながら文章を書いているので、読みにくい文章になっているし、最後まで読まずに、上から順に何も考えずにそのままプログラムを打ち込んだ人は私と同じ間違いを経験したことになり、申し訳ないことをしました。しかし、私のような本来は数学者で、正式のプログラミング教育を受けてない独学の人間がプログラムを作るとこんな感じでプログラミングしています。プロのプログラマーの人が見たら、全然関数を使わず、だらだら書いてある酷いプログラムだと思うと思います。私もパズルの解析をしたりするプログラムで複雑なアルゴリズムを表現しなければいけないときは良く考えてそれなりに綺麗なプログラムを作りますが、このような単純なプログラムはだらだらしたプログラムになります。兎も角、何でもいいから、でっち上げればいいです。バグの泥沼に嵌ったら、そこから抜け出すために自然とコメントのいっぱいついた綺麗なプログラムを書くようになります。多分、誰でもいっぱい間違えます。間違いを自分で直すことが出来るようになれば、プログラムが作れるようになります。実は、このプログラムは昔 Java で作った「ボヘミアン・シュナイダー」というトランプゲームのアプレット用のプログラムを最近 VC++ に翻訳したプログラムリストを見ながら作ったので、初めてのものを作るときのプログラミングより間違いは少なかったです。丸 1 日で作れました。500 行弱の誰でもが作れる長さのプログラムです。デバッギングが嫌で、プログラミングをあきらめる人が多いですが、その後に楽しい世界が待っています。トランプゲームを収集した書籍には松田道弘著「トランプゲーム辞典 ENCYCLOPEDIA of CARD GAME」東京堂出版や David Parlett 著「The Penguin Encyclopedia of Card Games」や赤桐裕二著「トランプゲーム大全」スモール出版があります。このような本は見つけたときに購入しておかないとすぐ絶版になります。最後の本は最近のものですが、最初の 2 冊は私は良い本だと思いますが今は絶版です。図書館か古本屋で探して見て下さい。この「ジャーマン・ホイスト」のプログラムを雛形にして、自分の気に入ったトランプゲームのプログラムを作ってください。そしてプログラムに知能を与えて見て下さい。自分が作ったものが思い通り動くと嬉しいものです。是非、挑戦してみて下さい。