Introduction

You've been asked by one of your friends to help him become better at playing 5 card stud poker; you know, the ole, “you know computers don't you?”. His first goal is to figure out the probability of each kind of hand. Reluctantly, you agree to help.

Playing Poker

Let's work with the basic 5 card stud poker.

Use a standard 52 card deck.

- Four Suits: Diamonds, Hearts, Clubs, Spades
- 13 cards in each suit: Ace, Two, Three, Four, Five, Six, Seven, Eight, Nine, Ten, Jack, Queen, King

The following are the definitions for each kind of hand:

- **Worthless**: Anything not listed below
- **Jack or Better**: 1 card of a Jack or higher
- **Pair**: 2 cards of the same rank (i.e. 2 Fives or 2 Nines)
- **Two Pair**: 2 pairs (i.e. 2 Threes and 2 Jacks)
- **Three of a Kind**: 3 cards of the same rank (i.e. 3 Fives or 3 Queens)
- **Straight**: 5 cards in a sequence (i.e. Eight, Nine, Ten, Jack, Queen)
- **Flush**: Any 5 cards of the same suit
- **Full House**: 2 cards of the same rank and 3 cards of the same rank
- **Four of a Kind**: 4 cards of the same rank (i.e. 4 Kings)
- **Straight Flush**: Combination of a Straight and a Flush
- **Royal Flush**: Ace, King, Queen, Jack, Ten of any suit

Deal 5 cards to each player (1 card at a time) and determine which player has the best “hand”.

Assignment

Write a program that computes the probabilities that each of the above kinds of hands might be dealt to a player. You should try this with 2, 3 or 4 players at a time and see if there is any difference in the probabilities with the different number of players.

The output for one run of your program might look something like this:

```
<table>
<thead>
<tr>
<th>Hand</th>
<th>Count</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>9624</td>
<td>9.6240%</td>
</tr>
<tr>
<td>JackOrBetter</td>
<td>40416</td>
<td>40.4160%</td>
</tr>
<tr>
<td>Pair</td>
<td>42356</td>
<td>42.3560%</td>
</tr>
<tr>
<td>TwoPair</td>
<td>4737</td>
<td>4.7370%</td>
</tr>
<tr>
<td>ThreeOfAKind</td>
<td>2143</td>
<td>2.1430%</td>
</tr>
<tr>
<td>Straight</td>
<td>363</td>
<td>0.3630%</td>
</tr>
<tr>
<td>Flush</td>
<td>192</td>
<td>0.1920%</td>
</tr>
<tr>
<td>FullHouse</td>
<td>148</td>
<td>0.1480%</td>
</tr>
<tr>
<td>FourOfAKind</td>
<td>15</td>
<td>0.0150%</td>
</tr>
<tr>
<td>StraightFlush</td>
<td>5</td>
<td>0.0050%</td>
</tr>
<tr>
<td>RoyalFlush</td>
<td>1</td>
<td>0.0010%</td>
</tr>
<tr>
<td>Total Hands</td>
<td>10000</td>
<td></td>
</tr>
</tbody>
</table>
```

Technical Details

- **Card** might (not required, but suggested) be a class that implements `IComparable.CompareTo` so that one card can be ranked against any other card. I used this to sort the cards in the hand to make it easier to search for the different hand types.
- **Deck** must (required) be a class that holds the cards. It should have at least the following methods. The parameters and etc to these methods are for you to make smart decisions.
  - `Initialize` (to build the deck)
  - `Shuffle` (to randomize the order of the cards)
  - `DealCard` (to remove a card from the deck)
- **PokerHand** might (not required, but suggested) be a class that holds the cards for a single player. It should have at least the following methods.
  - `ReceiveCard(Card c)` : to receive a card
- **PokerLogic** must (required) be a class that can examine a `PokerHand` (or whatever you choose to implement) to decide the hand type. This class would have the following kinds of methods (recommended, but not required)
  - `IsJackOrBetter(PokerHand h)`
  - `IsPair(PokerHand h)`
  - `...`
- You must (required) use enumerations (`enum`) to represent the cards & suits.
- Use the language features as best you can, this is not C++, this is C#, try to think accordingly.
NOTES:

- **Due: Monday, February 12**
- Include the heading required by your section at the top of your program
- Follow the style guidelines
- When complete submit your entire Visual Studio solution directory
- Submit through Eagle