Exercise 2

Dylan Sukha

Brief:

Create a computer program (for self) to play card games against the computer, being an intelligent opponent.

Card game(s) to be coded and played: *To be determined.*

UI/front-end to build for (web/native/console): *To be determined.*

Known Specifications + Attributes:

Needs:

- **Some implementation of a card deck.**
- **Cards**
- **A function to shuffle (assuming just shuffling the deck previously mentioned).**

Evaluating Front-End Interface:

Assuming C# will be the primary language for writing the program.

Web:
Interface can be composed with HTML, CSS and possibly supplementary JavaScript and PHP. C# can be used with ASP.NET. Web technologies are agile and cross platform. They are also easy to update without having to recompile and redeploy. Testing may be rigorous if the end product is intended for a variety of web browsers. Web Technologies can also be easily packaged to be used like native software.

Native:
XAML and API's like Win2D provide potential for rich aesthetics and have strong cohesion with C#. Native testing for a targeted operating system can be more deterministic from early on, keeping test cases manageable. If wanting to build for multiple platforms, porting code may cost considerable time. Ideal for building a final solution with little to no future maintenance (updates, bug fixes).

Console:
A command-line interface (CLI) is easy to implement in C# and very lightweight. For the context of card games, however, visual detail and user interaction is likely too little for an enjoyable experience.

Verdict: As someone building a card game(s) program for themselves, both *Native* and *Web* are viable options but I would choose *web* because I can start off building it for my intended device(s) and if I decide to release it to a wider audience in future, it would be easy due to its portable nature.

Initial List:

- **Create the C# solution – console program for initial prototype.**
- **Create a class for a Card object with some fields and properties.**
  - ***string for card typ. E.g. A,1,2,3...J,Q,K. Using a char would only occupy 1 byte of memory but can be troublesome in conversions so a string will be used.***
  - ***char for group. Uses only 1 byte of memory. Can be represented by { H, D, S, C }***
  - ***Optional Boolean, isFlipped. In case the game consists of hidden and exposed cards. Also just 1 byte of memory.***
- **Create a class for the Deck which manages a list of cards.**
  - ***Will have to call 'using System.Collections'.***
  - ***Will use a List. While a Stack would have a lower cost, C# does not natively have a shuffle function for its collections unlike Java.***
- **Write a *main* method to test the code.**
- **Later, determine a game, code UI basics and extend C# with ASP.NET.**

A sample solution has been completed. (exTwo.zip)

**~~~~~~**

Enjoy!