CS215: Introduction to Program Design, Abstraction and Problem Solving
(Spring, 2025)
Programming Assignment 2
(100 points)
Today's Date: Tuesday, April 1, 2025
Due Date: Sunday, April 20, 2025

Academic Honesty:
All assignments in class are individual work. All work submitted as part of the class must be your own. You may not share work, nor may you use code provided to you by others, except for your instructor. *You are allowed to use the source code provided by the instructor of this course only.*

Problem Statement

Write a program that plays a simple card game, named War (also known as Battle in the United Kingdom) (https://en.wikipedia.org/wiki/War_(card_game)), typically played by two players using a standard **52**-card deck. The objective of the game is to win all the cards and often played by children. The game is played as follows:

1. Each player gets dealt half the deck, **26** cards, and the cards are put face down in the pile in front of the players.
2. Both players turn their top card face up at the same time. The person with the higher card wins the draw and takes both the cards. They are put to the bottom of the pile, which the player can continue using cards on his/her pile. Aces are high, and suits are ignored.
3. If the two cards played are of equal value, then there is a "war". Both players place the next three cards face down and then another card face-up. The owner of the higher face-up card wins the "war" and adds all the cards on the table to the bottom of the winner's pile. If the face-up cards are again equal, then the battle repeats with another set of face-down/up cards. This repeats until one player's face-up card is higher than his/her opponent's or one player does not have enough cards to finish the war then loses immediately.
4. First player to finish all his/her cards loses the game.
5. If a player finishes his/her cards during a "war" without having enough cards to finish the "war" then loses immediately.

Part 1: Complete the definitions of the `Card` class and the `Deck` class for War Game

In OOP design, the classes should model things in the problem domain. As Part 1 of Project 2, you have defined TWO classes: the `Card` class (which models a single playing card of a standard 52-card game), and the `Deck` class (which models a standard 52-card deck), in Lab 9. Please note that after you complete the definitions of these two classes, they can be used not only for the War Game, but also other 52-card games.

Part 2: Complete the definition of Player class for War Game

As Part 2 of Project 2, after you get familiar with the definitions of classes named **Card** and **Deck**, you can start to work on the definition of the third class, named **Player**, which represents the pile of cards in one player's hand and the actions that a player may take during the War game, such as **play_a_card**; **addCards** when a player wins a round and gets all the cards on the table; **dropCards** when there is a tie, each player needs to drop 3 cards (face down) on the table, then play one more card (face up); and so on. The following shows the declaration of this class:

```
class Player
{
   public:
      // default constructor
      Player();

      // alternative constructor
      Player(vector<Card> ini_cards);

      // return how many cards player holds currently
      int getNumCards() const;

      // player plays one card from the front of cards at hand
      Card play_a_card();

      // when the player wins the round, this function will be called
      // player adds winning cards to the end of the cards at hand
      void addCards(vector<Card> winningCards);

      // when there is a tie, this function will be called
      // player drops THREE cards from the front of cards at hand
      vector<Card> dropCards();

      // display cards at player's hand
      void print() const;

      // you are allowed to add other member functions if you want

   private:
      int numCards;          // how many cards in player's hand
      list<Card> cards;      // sequence of cards in player's hand

};
```

In Lab10, you need to complete the definition of the class named **Player**.

Part 3: Provide your own main function to complete War Game

You can either based on the solution file named Lab10.sln in **Part 2** and change the file name of Lab10.cpp into Project2.cpp or create a new empty project, named Project2, then copy and add all source files you need to Project2 solution.

Start to write the main function to demonstrate the War game between two players:

1. Display one top card (suit and point) from each player, which represents the card played by each player in the current round
2. Display how many cards on the pile (on the table)
3. Decide which player wins the current round or it is a tie

    - If one player wins, display "Player x wins…get all cards from the pile!"
    - If it is a tie, display "Each player drops three cards (face down) on the pile, then play one more card (face up)"

4. Display how many cards are in player1's hand and how many cards in player2's hand
5. After each round, your program should ask the user "Do you want to continue…for the next round? (N or n to quit the game).

    - If the user clicks enter/return key, the game should continue to the next round, back to step 1
    - If the user clicks either "N" or "n" to stop the game, your program should display the following information, then quit. "You choose to quit the game! Player1 has XXX cards left! Player2 has YYY cards left!" where XXX and YYY are the number of cards in each player's hand at that moment respectively.

6. The first player to finish all his/her cards loses the game, and your program should stop and report who wins the game.
7. Your program should also stop immediately if one player finishes his/her cards during a "war" without having enough cards to finish the "war", then report who wins the game.
8. If both players finish cards at the same time, your program should report a tie game then stop.

Please download the following sample output file to test running your program, and especially check THREE testing cases described in the following pdf file:

http://www.cs.uky.edu/~yipike/CS215/WARGame_Samples.pdf

If you can demonstrate your Project 2 during Lab11 class, you may gain a maximum of 3 bonus points for Lab11.

Submission:

Open the link to course Canvas page (https://www.uky.edu/canvas/), and log in to your account using your linkblue user id and password. Please submit **TWO** files (**Project2.cpp** and **Project2.zip**) through the submission link for "**`Project 2`**". It is a good idea to check that your files have been uploaded successfully. If not, go back and submit again. If your submission does not contain the correct files, you lose points.

Note to zip the inside folder (project folder) (including all the files and sub-folders under it), you can simply right click the folder, and select **Sendto → Compressed (zipped) folder**, it will generate a zip file with the same name as the folder name by default. For example, if your project folder is called **Lab10**, then by default the zip file is called **Lab10.zip**; if your project folder is called **Project2**, then by default the zip file is called **Project2.zip** You can double check whether this zip file contains all header .h files and source .cpp files you need for Project 2, by double clicking the zip file. It should contain card.h, card.cpp, deck.h, deck.cpp, player.h, player.cpp and Project2.cpp (the main source file and also the solution file).

(Late assignment will be reduced 10% for each day that is late. The assignment will not be graded (you will receive zero) if it is more than 5 days late. Note that a weekend counts just as regular days. For example, if an assignment is due Friday and is turned in Monday, it is 3 days late.)
Always read the grading sheet for each project assignment. It lists typical errors. Check for these errors before submitting your source code. **Please note that your C++ program must compile to be graded. If your program cannot pass the compilation, you will get 0 point.**

(Grading sheet is shown on the next page!)

Grading Sheet for Project Assignment 2

Total: 100 points.

These are example errors. There are other ways to lose points. C++ programs must compile in order to be graded	Points	Deducted Points
Correctness	60	
Provide the correct main function to follow the description of the War game in the problem statement. You program repeatedly doing the following until the game is over either by the user or one player runs out of card:		
*Correctly display one top card (suit and point) from each player, which represents the card played by each player in the current round	5	
*Correctly display how many cards on the pile (on the table)	5	
*Correctly decide which player wins the current round or it is a tie	5	
*Correctly display how many cards in player1's hand and how many cards in player2's hand	5	
*After each round, your program should ask if the user wants to continue and take actions accordingly	5	
*First player to finish all his/her cards loses the game, and your program should stop and report who wins the game.	5	
* Your program should also stop immediately if one player finishes his/her cards during a "war" without having enough cards to finish the "war", then report who wins the game.	5	
Provide the correct implementation of member functions for Player class in Player.cpp	20	
Provide separate .cpp file and header file for class named Player	5	
Style	10	
Lay out your program in a readable fashion	3	
Include comments as specified in the lecture notes	4	
User-friendliness in I/O design	3	
Testing (No Documentation is required)	30	
Pass testing case 1 described in Sample output pdf file: **exactly match** the sample output under testing case 1;	10	
Pass testing case 2 described in Sample output pdf file: the user chooses to quit the program before the game is over and correctly report how many cards in each player's hand;	10	
Pass testing case 3 described in Sample output pdf file: your program needs to continue playing without the interaction with the user, and correctly decide which player wins the game or it is a tie game.	10	
Your Score		