ECE 175: Computer Programming for Engineering Applications

Final Project: *Hit me, 31 style!*

Due Date: Dec., 7th 2015, 8:00 AM, via D2L dropbox

A Game of 31

Terry Benedict just hired you to reprogram all the Blackjack machines at the *Villagio*. Your task is to develop a program that plays a game of 31. In particular, your program should satisfy the following requirements:

1. Playing cards are represented as variables of the following type:

```
typedef struct card_s {
      char suit[9];
      int value;
      struct card_s *pt;
} card;
```

   You are allowed to add attributes to this definition, but not to remove any.

2. The game is played using two decks of cards.
3. At the beginning, the user can choose to shuffle the deck or load a predefined sequence of cards from a file (for testing). The card values should be at a separate file containing only integers.
4. The deck is represented as a dynamic list of cards. The cards drawn from the deck are deleted from the list.
5. The player's hand is represented as a dynamic list of cards. The list is populated with the cards drawn by the player.
6. The dealer's hand is represented as a dynamic list of cards. The list is populated with the cards drawn by the dealer.
7. The player starts the game with $1,000. The game terminates if the player loses all his money.
8. The minimum bet is $20 and the maximum bet is $200. The system shall not allow the player to bet more than he has available.
9. The game rules are as follows:
   (a) The game proceeds in rounds.
   (b) Face cards (Kings, Queens, and Jacks) count as 10 points.
   (c) Aces count as 1 point or 11 points.
   (d) The player places his/her bet at the beginning of every round and before any cards are dealt.
   (e) Initially, the dealer and the player are dealt one card each, in the following order: `player-dealer.`
   (f) The card of the dealer is dealt face up.

(g) After the two cards are dealt, the player goes first. The player has the option of "hit" (being dealt additional an card) or "stand" (terminate his turn). The player's objective is to bring his/her hand as close to 31 points as possible. The player's turn terminates when the player stands, hits 31, or the player goes "bust" (goes over 31).

(h) If the player hits 14, he is given the option to "stand". The dealer has to hit 31 to win a hand for which a player has chosen to stand at 14.

(i) The dealer takes hits until his hand totals 27 or more points. The dealer cannot take a hit beyond 27.

(j) In case the player has hit 14, the dealer continues to draw cards until he hits 31 or goes bust.

(k) If the player does not go bust and his hand total is higher than the dealer's total or the dealer goes "bust", he/she wins the bet. If the dealer and the player have the same hand total, the bet is "pushed" (no win or loss). If the dealer has a higher hand than the player's hand, the player loses the bet.

(l) Exceptions are made if 14 is hit by one of the parties. If player hits 14 and dealer hits 31, the dealer wins. If dealer hits 14, he wins. If dealer hits 14, while the player has also hit 14, the dealer has to continuously draw to hit 31, in order to win.

(m) The deck is shuffled when less than 30 cards are left. The dealer's and the player's lists are deleted at the end of each round (memory is freed back to the system).

Optional Features for extra credit:

1. **The battle of the dealers.** Modify your code to continuously play a user-defined number of games automatically. The player is automated to play with the same rules as the dealer, except that the player hits a "soft 27", i.e., a 27 occurring when an ace is counted as 11.
2. **Game statistics.** Keep statistics on the number of games played, the player's winning percentage, the dealer's winning percentage, the percentage of hands equal to 27, and the percentage of player wins, when he hits 14.
3. **Game Variations.** Implement the functions of "double down," "split," and "insurance".
   (a) **Double down:** The player can opt to increase the initial bet by 100% in exchange for committing to stand after receiving exactly one more card (independent of the number of prior cards).
   (b) **Split:** If the first two cards dealt to the player have the same value, the player is allowed to split them into two hands, by doubling his/her initial bet. The dealer separates the two cards and draws a further card on each hand.
   (c) **Insurance.** If the dealer's face-up card is an ace, the player is offered the option of taking "insurance" before other cards are dealt. This is a side bet equal to the initial bet that is paid if the dealer gets a 31.
4. **Graphics.** Add graphics to your game. You can print cards using ascii art.

1 Administrative Details

Groups: You are allowed to form groups of two, or work individually. In case you form a group, you have to appear as a group to demo your code. Team members will be individually questioned on the *entire code*.

Honor code: You are expected to submit their own code. You may ask other for advice, and in general discuss the project, but you should WRITE YOUR OWN CODE. If any part of the code submitted by different students is identical, ALL involved parties will receive zero credit on the entire project and one letter reduction in their final grade. This policy will be very aggressively enforced. **ALL submitted code will be checked with a plagiarism detection tool** (e.g., `http://theory.stanford.edu/~aiken/moss/`).

Submit your code via D2L.

ECE175--Fall 2015
Grading Rubric for Final Project

	Criterion	Maximum Points	Exemplary (100%)	Proficient (75%)	Marginal (25%)	Unacceptable (0%)
	Requirements					
1	Deck shuffling	/10	Deck is implemented as a linked list. Deck is shuffled when less than 30 cards are left. The shuffling function produces a new sequence of cards per round and every time the program is restarted	Deck is shuffled but card sequences are not sufficiently unique	Deck is not shuffled when fewer than 30 cards are left	Deck is not implemented as a linked list and/or the deck is not shuffled
2	Card Dealing	/10	Cards are dealt in the proper order. Dealer's first card appears face up	The right number of cards is dealt, but not in a proper order	Cards are not dealt correctly	Card dealing is not implemented
3	Bet handling	/5	Bets are tallied correctly. Min and max bet are enforced	Bets are tallied correctly, but limits are not enforced	Bets are tallied incorrectly	Betting is not implemented
4	Dealer's hand	/10	Dealer's hand is impemented as a linked list. Memory is dynamically allocated and freed between game rounds	Dealer's hand is implemented as a dynamic list, but memory is not properly handled	Dealer's hand is not implemented as a dynamic list	Dealer's hand is not implemented
5	Player's hand	/10	Player's hand is impemented as a linked list. Memory is dynamically allocated and freed between game rounds	Player's hand is implemented as a dynamic list, but memory is not properly handled	Player's hand is not implemented as a dynamic list	Player's hand is not implemented
6	Game rules	/15	All game rules are followed	Most game rules are followed	Most game rules are not followed	Game is not functional
7	Game Interface	/10	The interface is inuitive. The user is able to play the game with minimal isntructions. Transitions from round to round are clear	The interface is intuitive for the most part. Some difficulty in understanding the game navigation.	The interface is counter-intuitive. Navigation options are not clearly stated. Interface limitations prevent proper game functionality	The interface is very basic and does not allow transitions between game rounds.
	Program Design					
8	Code modularity	/10	The code is logically divided to several functions that implement important functionality (deck shuffling, dealer hand, player hand, hand tallying, list creation, list deletion, etc.)	The code is modular but further simplification could have been attempted	The code only has a few functions. Most functionalities are integrated within the main function	Code is not modular

10	Code documentation	/5	The code is properly documented. The input/output and goal of every fucntion is adequately described. Comments are provided for various parts of the code	The code is partially documented	The code is only cursorily documented	No documentation is provided
11	Compilation	/15	Code succesfully compiles without errors or warnings. The code does not hang while in execution	The code succesfully compiles, but some conditions may make it hang	N/A	Code does not compile
Total		**/100**				

	Extra Credit					
12	Double Down	/3	Game allows the player to double down			
13	Split	/3	Game allows the player to spit on an identical pair			
14	Insurance	/3	Game allows the player to buy insurance when the dealer's face-up card is an ace			
15	Graphics	/4	Nice graphics are being used to print cards on screen			
16	Game statistics	/3	Game keeps track of game statistics. Statistics can be exported to a file			
17	The battle of the dealers	/4	Two dealers can play against each other a user-defined number of time. One of the dealers implements a soft hit at 17			
Total		**/20**				