

Computer Science 215
Tools and Techniques for Software Development
Fall 2001
Project 3 – Poker

Due: Monday, 10/22/2001 9:00 a.m.

For this assignment, you will write a program for the card game Poker. The program should be menu-driven as follows:

Please select one of the following options:

- 1 **Play**
- 2 **Set seed for deck**
- 3 **Shuffle cards**
- 4 **Read in a loaded deck**
- 5 **Print stats**
- 6 **Quit**

Option:

Upon start-up, the program should initialize the deck of cards and the statistics, and allow the user to enter options until Quit is selected. Numerical error checking for the menu item is required (i.e., make sure the value is in range). Information for each of the options follows after a brief description of the rules of Poker.

In the game of Poker, each player is dealt a hand of 5 cards. For our game, the player will be allowed to select cards for replacement two times before the round is over. The hand is evaluated according to the following rules (from highest to lowest hand):

- Royal Flush – A, K, Q, J, and 10 of the same suit
- Straight Flush – any five-card sequence of the same suit
- Four of a Kind – four cards of the same value
- Full House – a pair and three of a kind
- Flush – any five cards of the same suit
- Straight – five-card sequence, not of the same suit
- Three of a Kind – three cards of the same value
- Two Pair – two sets of two cards with the same value
- One Pair – one set of two cards with the same value
- High Card – the highest value card in hand

In our game, the player will be playing against a dealer, whose hand will be determined by probability. If there's a tie on type of hand, the player wins, except in the case of High Card – then the highest cards are compared.

A description of the menu options follows:

1 **Play** – When this option is selected, the user enters a playing mode. The user is dealt 5 cards from the current deck, which are displayed in sorted order on the screen with indices:

Current hand:

1: 3 D
2: 4 S
3: 9 H
4: 9 D
5: 9 C

The user is then asked for the number of cards to replace. After entering that number, the program prompts for the indices of those cards. If the user enters a 0, no cards will be replaced and the hand will be evaluated. If a 5 is entered, all cards will be re-dealt with no prompts. Note that the user gets two chances to receive new cards. After these dealings, the hand is evaluated and compared to the dealer's. A sample session follows, continuing from the previous example.

How many cards would you like to replace? (0..5) 2

Enter the index number of the card you'd like to replace (1..5): 1

Enter the index number of the card you'd like to replace (1..5): 2

Current hand:

1: 2 C
2: 9 H
3: 9 D
4: 9 C
5: J C

How many cards would you like to replace? (0..5) 2

Enter the index number of the card you'd like to replace (1..5): 1

Enter the index number of the card you'd like to replace (1..5): 5

Current hand:

1: A D
2: 9 D
3: 9 C
4: 9 H
5: Q S

Hand: Three of a Kind

Dealer's hand: High Card with high card Q D

You're a winner!!!

Current statistics:

Number of games: 1
Number of wins : 1

Winning % : 1.00

Would you like to play again? (y/n)

The dealer's hand is set randomly by calling the C function, `rand ()`, which will return an integer in the range [1..32767]. Your program will take that integer and assign the dealer's hand according to the following rules:

```
> 32000 royal flush
> 31000 straight flush
> 29000 four of a kind
> 27500 full house
> 25000 flush
> 23000 straight
> 20000 three of a kind
> 16000 two pair
> 10000 one pair
else high card
```

If the dealer and player both have high card hands, the values of the high card need to be compared to determine the winner. The player's high card is taken from his current hand, while the dealer's high card is the next card drawn from the deck.

Statistics are printed after each hand, at which time the cards are shuffled. Note that play continues until the user answers 'N' or 'n'.

2 Set seed for deck – Since your program depends on random numbers for shuffling, we need a way to make it play a different set of games each time the program is run. We can do this by seeding the random number generator an integer value. Note that by seeding the random number generator with a specific value, we will play the same set of games, which may be useful for testing your program. To get the seed, prompt the user for a seed value; otherwise, use the method provided on the website to get a semi-random seed.

3 Shuffle cards – This option allows the user to shuffle the deck of cards. Upon program start-up, the cards are not shuffled, so they must be shuffled at least once before play begins. When playing, the cards are shuffled after each hand. You should use the following algorithm to shuffle the cards:

```
for (each card)
    get a random number to use an index to swap the
        card at this index
    swap
```

4 Read in a loaded deck – This option will allow you to test certain hands, dealt from a loaded deck. The file should contain 52 lines, each with a numeric value (1..13), a space, and a single character for suit. Use the code from the website to help write this function. A sample deck with a royal flush is also available.

5 Print stats – This option will allow the user to print the statistics as given above, including the two lines of '*'s.

6 Quit – The program should terminate when this option is selected.

For this program, you **must** use **structs** and **typedefs** for the following types:

- CARD_T** – a struct containing two fields: an integer **value** and a character **suit** fields
- DECK_T** – a struct containing two fields: an array of **cards** and a pointer to the current top of the deck
- STATS_T** – a struct containing the number of wins and the number of games played

For this program, You **must** define functions with the **exact** prototypes listed below (these will *help* you):

```
void init_cards (DECK_T deck); /* initialize the 52 cards */
void init_stats (STATS_T *stats); /* initialize stats fields */
void play (DECK_T *deck, STATS_T *stats);
void set_seed ();
void read_deck (DECK_T *deck);
void shuffle_cards (DECK_T *deck); /* shuffles the deck */
void swap (CARD_T *p, CARD_T *q); /* swaps values of two cards */
void deal_cards (CARD_T *hand, DECK_T *deck, int num_cards);
void sort_hand (CARD_T hand []);
int replace_cards (CARD_T hand [], DECK_T *deck);
void print_stats (STATS_T *stats); /* prints the stats */
HAND_T get_dealer_hand ();
HAND_T get_hand_type (CARD_T hand []);
```

play is the main playing function; it is not exited until the user enters 'n' or 'N'.

set_seed prompts the user and initializes the random number generator.

read_deck reads in the loaded deck and stores it in **deck**.

deal_cards assigns the first **n** cards from the **deck** to **hand**.

sort_hand sorts the cards by value in **hand**.

replace_cards prompts the user for the number of cards to replace, then for the indices, and replaces the card in **hand** at each index entered with the card from the top of the **deck**. It returns the number of cards replaced.

get_dealer_hand determines the dealer's hand from the probability values given above and returns it.

get_hand_type determines the player's hand given in **hand** and returns it.

You may also find the following functions useful:

```
int menu (); /* prints menu and gets option */
void print_card (CARD_T card); /* prints a single card */
void print_hand (CARD_T hand []); /* prints the hand */
```

On the web page, you will find files called **poker_hands.c**, **poker_hands.h**, and **poker.h** with some partial code that you need to complete. You must also keep these files separate and compile and link them. Your program should be called **poker.c** and should include any .h files it needs. Compile the program as follows:

```
cc poker.c poker_hand.c -o poker
```