

Forum:[Hack Forums](#)

Thread: [<C Homework> Create a Struct Card Deck and Function to Shuffle](#)

Mad Hatter - November 20, 2016, 2:05 pm

Hey guys, I'm doing my homework for a C class I'm taking and this weeks assignment is to create a structure (struct card) that will represent a card in a standard deck of playing cards. I need to represent both the suit (clubs, diamonds, hearts or spades) as well as the rank (A, K, Q, J, 10, 9, 8, 7, 6, 5, 6, 7, 2) of each card. I also need to write a function that will perform a perfect shuffle on a deck of cards represented using the data structures from the deck. In a perfect shuffle, the deck is broken exactly in half and rearranged so that the first card is followed by the 27th card, followed by the second card, followed by the 28th card, and so on.

I'm really confused on how to do this. I'm pretty sure you'd make the struct deck like so:

Code struct deck{

```
char suit[10];  
char rank;
```

```
};
```

But, as far as creating all 52 cards and assigning correct suits and ranks to all card and making a function to perform a perfect shuffle. I'm kind of lost. If you don't want to help me out for free I completely understand, if you know how to do this and would like to get compensated for it, please PM me and I'll give you a few bucks via Paypal for helping me. Code must be original. Thanks!

Kantos - November 20, 2016, 9:35 pm

I would suggest not making your deck struct like that. Really, a card doesn't have to be a struct, you can get all the info about a card just by assigning each card an integer from 0 to 51 called C and determining the suit and rank from C.

You get the suit of a card C by calculating $C \% 4$ (the modulus operator, the remainder after dividing C by 4) and the rank with $C \% 13$.

So really your deck of cards can just be an int array, i.e. `int deck[52]` where you store the C for each card and then you can make a separate `print_card©` function since you can get the suit and rank from C.

If you want to initialize the deck, you just do something like

```
Code int i;  
for(i = 0; i < 51; i++)  
    deck[i] = i;
```

This will arrange the deck like this:

```
0 0 0  
1 1 1  
2 2 2  
3 3 3  
4 0 4  
5 1 5
```

```
6 2 6
7 3 7
8 0 8
9 1 9
10 2 10
11 3 11
12 0 12
13 1 0
14 2 1
```

Where the first column is C, the second is the suit, and the third is the rank. So it arranges the deck in a non-standard manner (the usual way would be by suit). If it needs to be sorted by suit, then I can teach you how you can do that using `qsort()`.

Alternatively you could make an array of Card structs that just hold two integers, suit and rank, and then calculating C by doing `suit*13 + rank`. C will be ordered by suit naturally as well.

To do the perfect shuffle, you're basically doing a merge operation between the first half and second half of the deck. The easiest way would be to create a separate array to store the shuffled deck, and then create a loop that takes the first and 27th cards, the second and 28th cards, etc. and copies it to the shuffled array. Very straightforward.

0xDEAD10CC - November 20, 2016, 11:07 pm

Why are you using a string to represent the suit? What's wrong with enums? And using a struct over a random integer array is much easier to read too. I don't see the benefit in using a function like `qsort()` here either when the cards in a standard deck are pretty well known. Unless you've divided the deck up, a simple arrangement of for loops will be much better anyways to initialize it to the desired manner, and more efficient too.

Here's a start:

```
Code #ifdef _MSC_VER
# pragma warning(disable: 4996)
#endif

#include <stdio.h>

#define UNUSED(x) (void)(x)

enum card_suit { HEARTS, DIAMONDS, SPADES, CLUBS };

struct card
{
    enum card_suit suit;
    int value;
};

const char *suit_name(const struct card *card)
{
    switch (card->suit)
    {
        case HEARTS: return "Hearts";
        case DIAMONDS: return "Diamonds";
        case SPADES: return "Spades";
```

```

        case CLUBS: return "Clubs";
        default: return "INVALID"; /* or Joker? but this requires 54 cards */
    }
}

```

```

void print_card(const struct card *card)
{
    static const char *values[13] =
    {
        "Two", "Three", "Four", "Five", "Six",
        "Seven", "Eight", "Nine", "Ten", "Jack",
        "Queen", "King", "Ace"
    };

    printf("%s of %s\n", values[card->value - 2], suit_name(card));
}

```

```

void init_cards(struct card cards[52])
{
    int i;
    for (i = 0; i < 13; ++i)
    {
        cards[i].suit = HEARTS;
        cards[i].value = i + 2;

        cards[i + 13].suit = DIAMONDS;
        cards[i + 13].value = i + 2;

        cards[i + 26].suit = SPADES;
        cards[i + 26].value = i + 2;

        cards[i + 39].suit = CLUBS;
        cards[i + 39].value = i + 2;
    }
}

```

```

int main(int argc, const char *argv[])
{
    int i;
    struct card cards[52];
    init_cards(cards);

    for (i = 0; i < 52; ++i)
        print_card(cards + i);

    UNUSED(argc);
    UNUSED(argv);
    return 0;
}

```

And this will demonstrate sorting by suit just fine without using a blind sorting method like qsort(). No need for that, we know that there's 4 suits, and 52 cards in a deck, which means 13 cards of each suit.

REDRIFLE - *November 23, 2016, 5:26 pm*

Colleges need to step it up, these kinds of programs should be taught in HS.