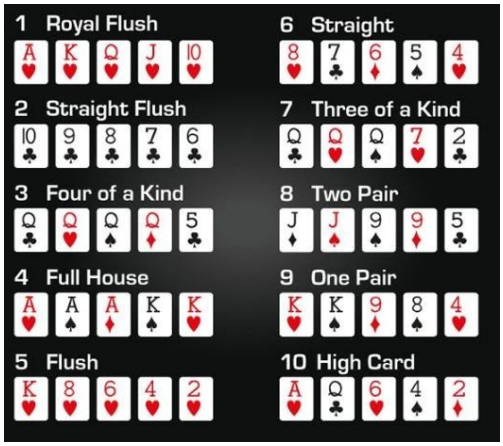
Lab7: Poker

How to play poker:
Each player gets dealt 5 cards. Each player decides which cards to keep, and which cards to replace. Players discard between 0-5 cards, and are dealt that same number of new cards. Players then show their hands to see who has the highest poker hand.

Poker hands:
Single card. High card wins.
Pair. High pair wins.
2 Pairs. Highest pair wins. If those tie, second highest pair wins.
3 of a kind. 3 of the same card.
Straight. 5 cards in a row. A2345 is the lowest, 10JQKA is the highest.
Flush. 5 cards of the same suit.
Full house. A 3 of a kind and a pair.
4 of a kind. 4 of the same card.
Straight flush. 5 cards in a row of the same suit.



Implementation:
A card will be represented by an int from 1-52.

Card	Suit	Face
1	S	A
2	S	2
...		
11	S	J
12	S	Q
13	S	K
14	H	A
15	H	2
...		
24	H	J
25	H	Q
26	H	K

Card	Suit	Face
27	C	A
28	C	2
...		
37	C	J
38	C	Q
39	C	K
40	D	A
41	D	2
...		
50	D	J
51	D	Q
52	D	K

A deck will be represented by a 52 element int array.

Lab7: Poker

Part 1: Helper Functions

Task 1: Create a function that takes an int card (1-52), and returns its face. You should return an int from 1-13 where:
1 = A, 2 = 2, 3 = 3, ... , 10 = 10, 11 = J, 12 = Q, 13 = K.
Tip: Use int / mod division.
public static int face(int card)

```
//Insert Code with comments here:
```

Task 2: Create a function that takes an int card (1-52), and returns its suit. You should return an int from 1-4 where:
1 = Spade, 2 = Heart, 3 = Clover, 4 = Diamond
Tip: Use int / mod division.
public static int suit(int card)

```
//Insert Code with comments here:
```

Task 3: Create a function that takes an int card (1-52), and returns a nice looking String.
Use A, J, Q, K for face. Because we want cards to be 2 letters, the 10 presents a problem. Use X instead of 10. Ex, heart ace = "HA", diamond king = "DK", spade 3 = "S3", clover 10 = "CX"
public static String cardToString(int card)

```
//Insert Code with comments here:
```

Task 4: Create a function that takes an int[] of cards, and returns a nice looking String. If the int[] is null, return "[]". This function should be used for both decks (the stack of 52 cards you're playing with) and hands (each player's 5 cards).
Ex, handToString({1,16,40}) => "[SA, H3, DA]"
public static String cardsToString(int[] cards)

```
//Insert Code with comments here:
```

Task 5: Create a function that takes a 52 element int[] of cards called deck, and shuffles it. Our shuffling algorithm will be as follows:

Swap each card in the deck with another card at a random index. A card may be swapped with itself.

Because we are shuffling (changing) the deck itself, we dpublic static void shuffle(int[] deck)
on't need to return it.

```
//Insert Code with comments here:
```

Lab7: Poker

Task 6: Create a function that creates a 52 element shuffled int[] of cards.
public static int[] createShuffledDeck()

```
//Insert Code with comments here:
```

Task 7: Create a function that returns the next card in the deck. As we "use" cards, we should set their spot in the deck to 0 to represent empty. You should use cards in index order, meaning that as you're using cards, the deck with have 0's at the front. If you run out of cards in the deck, return 0.
public static int nextCard(int[] deck)

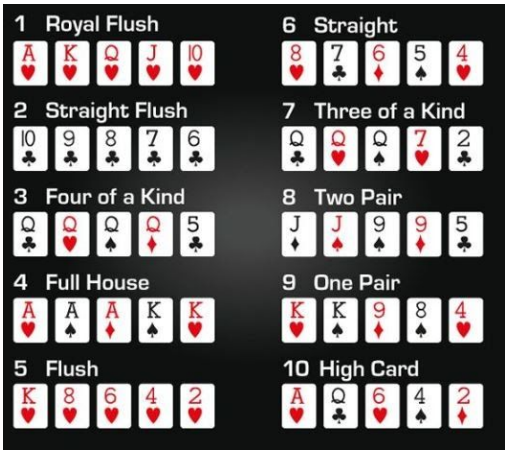
```
//Insert Code with comments here:
```

Task 8: Create a function called task8() and follow these steps:

1. Create a shuffled deck.
2. Print the deck.
3. Create a for loop that prints the first 5 cards in the deck, 1 card per line.
4. Look at the hand to determine its best poker value. Paste the output into the submission box, and indicate its best poker value as a comment.

public static void task8()

```
//Insert Code with comments here:
```



Task 9: Create a function that takes a 5 element int[] of cards called hand, and returns its best poker hand using the values in the picture. You don't need to worry about the fact that a pair of 10's beats a pair of 9's, and we can just say that those 2 hands tie.
public static int bestPokerHand(int[] hand)

```
//Insert Code with comments here:
```

Lab7: Poker

Task 10: Create a function that takes a poker hand called hand as an int from 1-10, and returns the name of that poker hand. Ex, pokerHand(1) => "Royal Flush. pokerHand(9) -> "One Pair".
public static String pokerHand(int hand)

```
//Insert Code with comments here:
```

Task 11: Create a play function that follows these steps:

1. Create a shuffled deck of 52 cards.
2. Create 2 5 element hands called handA, handB.
3. Deal cards into each hand in alternating order. IE, give a card to handA, give a card to handB, give a card to handA, etc... Yes, it must be alternating.
4. Display "Player 1's turn" and show them their hand.
5. Ask them which card they want to replace, using numbers 1-5. If they don't want to replace a card, they should type in 0.
6. Since users are allowed to replace more than 1 card, keep asking them this question until they enter 0.
7. Only after they've indicated all the cards to replace, go through and replace those cards with new cards.
8. Display the new hand.
9. Repeat steps 4-8 for Player 2.
10. Determine the winner and print either "Player 1 Wins!" or "Player 2 Wins!"

public static void play()

```
//Insert Code with comments here:
```