```
1 /*****************************************************************************
2  * PROGRAMMER : Ali Eshghi
3  * STUDENT ID : 1112261
4  * CLASS      : CS1C
5  * SECTION    : MW 5pm
6  * Assign #1  : Deck of cards
7  * DUE DATE   : 22 January 2020
8  *****************************************************************************/
9 #ifndef HEADER_H_
10 #define HEADER_H_
11
12 //Preprocessor directives
13
14 #include<iostream>   //for input,  output
15 #include<iomanip>    //for output style
16 #include<stdlib.h>   //for srand, rand
17 #include<time.h> //for time
18
19 using namespace std; //using namespace standard
20
21
22 const int DECK_SIZE = 52;
23
24 struct Card
25 {
26     string suit;//(spades, diamonds, hearts, clubs)
27     string rank;//(Ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, jack, queen, king)
28
29 };
30
31
32 class Deck
33 {
34 public:
35     //default constructor: assigning the 52 cards
36     Deck();
37
38     //destructor
39     ~Deck();
40
41     //initializing the deck of cards
42     void Initialize();
43
44     //shuffle the deck when the cards are assign
45     void shuffle();
46
47     //print the current deck with the current order
48     void print() const;
49
50     //compares the two decks
51     bool compare() const;
52
53     //method for stating how many perfect shuffle
54     //needed to return the deck to original
55     void returnToOriginal() const;
56
57 private:
58
59     //variable Deck to the pointer of deck
```

```
60      Card deck[DECK_SIZE];
61      Card shuffled[DECK_SIZE];
62      int  shuffleCount;
63
64  };
65
66
67  void PrintHeader (string asName, // IN - assignment Name
68                    int    asNum);    // In - assignment number
69
70  #endif /* HEADER_H_ */
71
```

```cpp
/*******************************************************************************
 * PROGRAMMER : Ali Eshghi
 * STUDENT ID : 1112261
 * CLASS      : CS1C
 * SECTION    : MW 5pm
 * Assign #1  : Deck of cards
 * DUE DATE   : 22 January 2020
 *******************************************************************************/

#include"Header.h"

/*****************************************************************
 * Deck of Cards
 * ______________________________________________________________
 * This program initializes a new deck of cards and does a
 * perfect shuffle on the card and prints out the original
 * deck and the shuffled deck, then it calculates how many
 * more perfect shuffle is needed for the shuffled deck to
 * go back to original deck.
 * ______________________________________________________________
 * INPUT: N/A
 *
 * OUTPUT: original deck of card, shuffled deck of card,
 *         the final deck of cards, the number of perfect
 *         shuffles needed to get back to original.
 *
 *
 *****************************************************************/

int main()
{
    //Constants
     const string asName = "Deck of Cards"; //assignment name
     const int    asNum  = 1;               //assignment number


    //Variables

    bool    compare;//PROCESS - boolean variable for compare
    Deckcard;       //PROCESS - class type3 variable for deck of card


    //printing the header file
    PrintHeader(asName, asNum);

    //initializing the deck of cards
    card.Initialize();

    //printing the deck
    card.print();

    //shuffling the cards
    card.shuffle();

    //printing the deck of card
    card.print();

    //comparing the deck of cards
    compare = card.compare();
```

```
60
61
62      //while loop to shuffle the cards until the shuffled deck is like
63      //the original deck
64      while(compare == false)
65      {
66          card.shuffle();
67          compare = card.compare();
68      }
69
70      //printing the final deck
71      if(compare == true)
72      {
73          card.print();
74          card.returnToOriginal();
75      }
76
77      //returning the 0 for program that eliminates successfully
78      return 0;
79  }
80
```

```cpp
/*****************************************************************************
 * PROGRAMMER : Ali Eshghi
 * STUDENT ID : 1112261
 * CLASS      : CS1C
 * SECTION    : MW 5pm
 * Assign #1  : Deck of cards
 * DUE DATE   : 22 January 2020
 *****************************************************************************/

#include"Header.h"


/*****************************************************************************
 * Methods for class Deck
 *****************************************************************************/


/*************************************************************
 * Deck();
 * Constructor: initializes the count of the shuffles
 * Parameters: shuffleCount
 * Return: none
 *************************************************************/
Deck::Deck()
{
    shuffleCount = 0;
}


/*************************************************************
 * ~Deck();
 * Constructor: performs nothing
 * Parameters: none
 * Return: none
 *************************************************************/
Deck::~Deck() {}

/*************************************************************
 * void Initialize();
 * This method will initialize a new deck of cards.
 -------------------------------------------------------------
 * Parameter:
 -------------------------------------------------------------
 * Return: none
*************************************************************/
void Deck::Initialize()
{
    int index;

    //all the face values in an string array
    string faces[] = {"Ace", "2", "3", "4", "5", "6", "7", "8", "9", "10", "Jack",
"Queen", "King"};

    //put all the suit values in an array as strings
    string suits[] = {"Hearts", "Diamonds", "Clubs", "Spades"};

    for(int i = 0; i < 4; i++)
    {
        for(int j = 0; j < 13; j++)
```

```
        {
            index = (i * 13) +j;

            deck[index].suit = suits[i];
            deck[index].rank = faces[j];
        }
    }
}


/*****************************************************************
 * void Print() const;
 * This method will print deck of cards.
 ----------------------------------------------------------------
 * Parameter:
 ----------------------------------------------------------------
 * Return: none
*****************************************************************/
void Deck::print() const
{
    for(int i = 0; i < DECK_SIZE; i++)
    {
        cout << deck[i].rank << " of " << deck[i].suit << endl;
    }

    cout << endl << endl;

}

/*****************************************************************
 * void shuffle();
 * This method will shuffle the deck of cards.
 ----------------------------------------------------------------
 * Parameter:
 ----------------------------------------------------------------
 * Return: none
*****************************************************************/
void Deck::shuffle()
{
    shuffleCount++;

    Card temp[DECK_SIZE];

    for(int i = 0; i < DECK_SIZE; i += 2)
    {
        temp[i] = deck[i/2];
    }

    for(int j = 1; j < DECK_SIZE; j += 2)
    {
        temp[j] = deck[j/2 + DECK_SIZE/2];
    }

    for(int k = 0; k < DECK_SIZE; k++)
    {
        shuffled[k] = temp[k];
        deck[k] = temp[k];
    }

```

```
118 }
119
120 /*****************************************************************
121  * bool compare();
122  * This method will compare the two deck of cards
123 -----------------------------------------------------------------
124  * Parameter:
125 -----------------------------------------------------------------
126  * Return: cmpr (bool type variable)
127 *****************************************************************/
128 bool Deck::compare() const
129 {
130     bool cmpr;
131
132     Deck card;
133
134     card.Initialize();
135
136     for(int i = 0; i < DECK_SIZE; i++)
137     {
138         if((deck[i].suit == shuffled[i].suit)
139         &&  deck[i].rank == shuffled[i].rank)
140         {
141             cmpr = true;
142         }
143
144         else
145         {
146             cmpr = false;
147         }
148
149     }
150
151     return cmpr;
152 }
153
154 /*****************************************************************
155  * void returnToOriginal();
156  * This method will prints out how many shuffles needed to
157 -----------------------------------------------------------------
158  * Parameter:
159 -----------------------------------------------------------------
160  * Return: none
161 *****************************************************************/
162 void Deck::returnToOriginal() const
163 {
164     cout << shuffleCount << "  shuffles needed to return the deck to "
165          << "its original form" << endl;
166 }
167
168
169
```

```cpp
/*****************************************************************************
 * PROGRAMMER : Ali Eshghi
 * STUDENT ID : 1112261
 * CLASS      : CS1C
 * SECTION    : MW 5pm
 * Assign #1  : Deck of cards
 * DUE DATE   : 22 January 2020
 *****************************************************************************/

#include"Header.h"

void PrintHeader (string asName, // In - assignment Name
                  int asNum)     // In - assignment number
{
    char asType = 'A';

    //OUTPUT - Console

    cout << left;

    cout << "*************************************************\n";
    cout << "* PROGRAMMED BY : Ali Eshghi" << endl;
    cout << "* " << setw(14) << "STUDENT ID" << ": 1112261" << endl;
    cout << "* " << setw(14) << "CLASS" << ": CS1B - MW - 7pm"<<
            endl;
    cout << "* ";

    // PROCESSING - This will adjust setws and format appropriately
    // based on if this is a lab 'L' or assignment.


    if (asType == 'L')
    {
        cout << "LAB #:" << setw(9);
    }

    else
    {
        cout << "ASSIGNMENT #" << setw(2);
    }

    cout << asNum << ": " << asName << endl;
    cout <<
            "*************************************************\n\n";
    cout << right;

    }
```

```
 1 ***************************************************
 2 * PROGRAMMED BY : Ali Eshghi
 3 * STUDENT ID    : 1112261
 4 * CLASS         : CS1B - MW - 7pm
 5 * ASSIGNMENT #1 : Deck of Cards
 6 ***************************************************
 7
 8 Ace of Hearts
 9 2 of Hearts
10 3 of Hearts
11 4 of Hearts
12 5 of Hearts
13 6 of Hearts
14 7 of Hearts
15 8 of Hearts
16 9 of Hearts
17 10 of Hearts
18 Jack of Hearts
19 Queen of Hearts
20 King of Hearts
21 Ace of Diamonds
22 2 of Diamonds
23 3 of Diamonds
24 4 of Diamonds
25 5 of Diamonds
26 6 of Diamonds
27 7 of Diamonds
28 8 of Diamonds
29 9 of Diamonds
30 10 of Diamonds
31 Jack of Diamonds
32 Queen of Diamonds
33 King of Diamonds
34 Ace of Clubs
35 2 of Clubs
36 3 of Clubs
37 4 of Clubs
38 5 of Clubs
39 6 of Clubs
40 7 of Clubs
41 8 of Clubs
42 9 of Clubs
43 10 of Clubs
44 Jack of Clubs
45 Queen of Clubs
```

```
46 King of Clubs
47 Ace of Spades
48 2 of Spades
49 3 of Spades
50 4 of Spades
51 5 of Spades
52 6 of Spades
53 7 of Spades
54 8 of Spades
55 9 of Spades
56 10 of Spades
57 Jack of Spades
58 Queen of Spades
59 King of Spades
60
61
62 Ace of Hearts
63 Ace of Clubs
64 2 of Hearts
65 2 of Clubs
66 3 of Hearts
67 3 of Clubs
68 4 of Hearts
69 4 of Clubs
70 5 of Hearts
71 5 of Clubs
72 6 of Hearts
73 6 of Clubs
74 7 of Hearts
75 7 of Clubs
76 8 of Hearts
77 8 of Clubs
78 9 of Hearts
79 9 of Clubs
80 10 of Hearts
81 10 of Clubs
82 Jack of Hearts
83 Jack of Clubs
84 Queen of Hearts
85 Queen of Clubs
86 King of Hearts
87 King of Clubs
88 Ace of Diamonds
89 Ace of Spades
90 2 of Diamonds
```

```
 91 2 of Spades
 92 3 of Diamonds
 93 3 of Spades
 94 4 of Diamonds
 95 4 of Spades
 96 5 of Diamonds
 97 5 of Spades
 98 6 of Diamonds
 99 6 of Spades
100 7 of Diamonds
101 7 of Spades
102 8 of Diamonds
103 8 of Spades
104 9 of Diamonds
105 9 of Spades
106 10 of Diamonds
107 10 of Spades
108 Jack of Diamonds
109 Jack of Spades
110 Queen of Diamonds
111 Queen of Spades
112 King of Diamonds
113 King of Spades
114
115
116 Ace of Hearts
117 2 of Hearts
118 3 of Hearts
119 4 of Hearts
120 5 of Hearts
121 6 of Hearts
122 7 of Hearts
123 8 of Hearts
124 9 of Hearts
125 10 of Hearts
126 Jack of Hearts
127 Queen of Hearts
128 King of Hearts
129 Ace of Diamonds
130 2 of Diamonds
131 3 of Diamonds
132 4 of Diamonds
133 5 of Diamonds
134 6 of Diamonds
135 7 of Diamonds
```

```
8 of Diamonds
9 of Diamonds
10 of Diamonds
Jack of Diamonds
Queen of Diamonds
King of Diamonds
Ace of Clubs
2 of Clubs
3 of Clubs
4 of Clubs
5 of Clubs
6 of Clubs
7 of Clubs
8 of Clubs
9 of Clubs
10 of Clubs
Jack of Clubs
Queen of Clubs
King of Clubs
Ace of Spades
2 of Spades
3 of Spades
4 of Spades
5 of Spades
6 of Spades
7 of Spades
8 of Spades
9 of Spades
10 of Spades
Jack of Spades
Queen of Spades
King of Spades


8  shuffles needed to return the deck to its original form

```