Khoi Duong

Prof. Yang

CS360

7/7/2022

HW#3

1.

Card.h:

Github link: https://github.com/MynameisKoi/CS360/blob/main/Card.h

```
// Card.h
// Card class definition.

#ifndef CARD_H
#define CARD_H

#include <string>
using namespace std;

class Card{
    private:
        int face, suit;
    public:
        static const std::string faces[];
        static const std::string suits[];
    Card();
    Card(int suit, int face);
    virtual ~Card();
    std::string toString();
};

#endif // CARD_H
```

DeckOfCards.h:

https://github.com/MynameisKoi/CS360/blob/main/DeckOfCards.h

```
// DeckOfCards.h
// DeckOfCards class definition.

#ifndef DECKOFCARDS_H
#define DECKOFCARDS_H

#include "Card.h"

class DeckOfCards{
    private:
        Card deck[52];
        int currentCard;
    public:
        DeckOfCards();
        void shuffle();
        Card dealCard();
        bool moreCards();
};

#endif // DECKOFCARDS_H
```

Main.cpp:

Github link: https://github.com/MynameisKoi/CS360/blob/main/hw%233_1.cpp

```
#include <iostream>
#include <iomanip>
#include "Card.h"
#include "DeckOfCards.h"
#include <stdlib.h>

using namespace std;

// Card.cpp
const std::string Card::faces[] = {"Ace", "1", "2", "3", "4", "5", "6", "7", "8",
"9", "10", "Jack", "Queen", "King"};
const std::string Card::suits[] = {"Clubs", "Diamonds", "Hearts", "Spades"};

Card::Card()
{
    face = 0;
    suit = 0;
}
```

```cpp
Card::Card(int suit, int face){
    this->face = face;
    this->suit = suit;
}

Card::~Card()
{
    //destructor
}

std::string Card::toString(){
    return Card::faces[face] + " of " + Card::suits[suit];
}

// DeckOfCards.cpp
// Default constructor
DeckOfCards::DeckOfCards()
{
    currentCard = 0;
    for (int suit = 0; suit < 4; suit++)
    {
        for (int face = 0; face < 13; face++)
        {
            deck[suit*13+face] = Card(suit, face);
        }
    }
}

// Shuffle cards
void DeckOfCards::shuffle()
{
    for (int i = 0; i < 52; i++)
    {
        int j = rand() % 52;
        Card temp = deck[i];
        deck[i] = deck[j];
        deck[j] = temp;
    }
}

// Deal card
Card DeckOfCards::dealCard()
{
    return deck[currentCard++];
}
```

```
// Check if there are more cards
bool DeckOfCards::moreCards()
{
    return currentCard < 52;
}

// main.cpp
int main(){
    DeckOfCards deck;
    deck.shuffle();

    int j = 0;
    while(deck.moreCards()){
        Card card = deck.dealCard();
        cout << card.toString() << endl;
    }

    if (++j % 4 == 0) {
        j = 0;
        cout << endl;
    }
    return 0;
}
```

Run program & result:

```
PS D:\VS CODE\C C++\CS360\HW#3>
} ; if ($?) { .\1 }
2 of Spades
8 of Clubs
9 of Diamonds
9 of Clubs
4 of Diamonds
7 of Diamonds
Queen of Diamonds
4 of Hearts
Jack of Spades
Jack of Clubs
9 of Hearts
1 of Diamonds
2 of Hearts
8 of Spades
5 of Spades
3 of Spades
10 of Spades
3 of Clubs
Queen of Hearts
7 of Clubs
9 of Spades
Jack of Hearts
3 of Diamonds
Ace of Spades
6 of Clubs
1 of Spades
7 of Spades
2 of Diamonds
1 of Hearts
8 of Diamonds
1 of Clubs
10 of Hearts
3 of Hearts
Queen of Spades
5 of Clubs
4 of Spades
8 of Hearts
6 of Spades
2 of Clubs
Jack of Diamonds
10 of Diamonds
5 of Diamonds
7 of Hearts
Ace of Clubs
6 of Hearts
```

2.

IntegerSet.h:

Github link: https://github.com/MynameisKoi/CS360/blob/main/IntegerSet.h

```
// IntegerSet.h
// IntegerSet class definition.

#ifndef INTEGERSET_H
#define INTEGERSET_H

class IntegerSet{
    private:
        const int max = 100;
        bool arr[100];
    public:
        IntegerSet(); // default constructor
        IntegerSet(int array[], int size); // constructor with array of integers
        IntegerSet unionOfSets(IntegerSet other); // union of two sets
        IntegerSet intersectionOfSets(IntegerSet other); // intersection of two
sets
        void insertElement(int k);
        void deleteElement(int k);
        void printSet();
        bool isEqualTo(IntegerSet other);
};

#endif // INTEGERSET_H
```

Main.cpp:

Github link: https://github.com/MynameisKoi/CS360/blob/main/hw%233_2.cpp

```
#include <iostream>
#include <iomanip>
#include "IntegerSet.h"

using namespace std;

IntegerSet::IntegerSet()
{
```

```
    for (int i = 0; i < max; i++)
    {
        arr[i] = false;
    }
}

IntegerSet::IntegerSet(int array[], int size)
{
    for (int i = 0; i < max; i++)
    {
        arr[i] = false;
    }
    for (int i = 0; i < size; i++)
    {
        arr[array[i]] = true;
    }
}

IntegerSet IntegerSet::unionOfSets(IntegerSet other)
{
    IntegerSet result;
    for (int i = 0; i < max; i++)
    {
        if (this->arr[i] || other.arr[i])
        {
            result.arr[i] = true;
        }
    }
    return result;
}

IntegerSet IntegerSet::intersectionOfSets(IntegerSet other)
{
    IntegerSet result;
    for (int i = 0; i < max; i++)
    {
        if (this->arr[i] && other.arr[i])
        {
            result.arr[i] = true;
        }
    }
    return result;
}

void IntegerSet::insertElement(int k)
```

```
{
    arr[k] = true;
}

void IntegerSet::deleteElement(int k)
{
    arr[k] = false;
}

void IntegerSet::printSet()
{
    int count = 0;
    for (int i = 0; i < max; i++)
    {
        if (arr[i] == true)
        {
            cout << i << " ";
        }
        else
        {
            count += 1;
        }
    }
    if (count == max)
    {
        cout << "None";
    }
    cout << endl;
}

bool IntegerSet::isEqualTo(IntegerSet other)
{
    for (int i = 0; i < max; i++)
    {
        if (this->arr[i] != other.arr[i])
        {
            return false;
        }
    }
    return true;
}

int main(){
    IntegerSet set1;
    IntegerSet set2;
```

```
    int array1[10] = {29, 14, 10, 8, 15, 20, 12, 18, 19, 21};
    IntegerSet set3(array1, 10);
    IntegerSet set4(array1, 10);
    int array3[5] = {3, 8, 7, 10, 12};
    IntegerSet set5(array3, 5);

    set1.insertElement(29);
    set1.insertElement(10);
    set1.insertElement(12);
    set1.insertElement(18);
    cout << "Set 1: ";
    set1.printSet();

    set2.insertElement(1);
    set2.insertElement(2);
    set2.insertElement(3);
    set2.insertElement(4);
    set2.insertElement(5);
    cout << "Set 2: ";
    set2.printSet();
    set2.deleteElement(3);
    set2.deleteElement(5);
    cout << "Set 2 after deletion: ";
    set2.printSet();

    cout << "Set 3: ";
    set3.printSet();
    cout << "Set 4: ";
    set4.printSet();
    cout << "Set 5: ";
    set5.printSet();

    cout << "Set 1 union Set 4: ";
    set1.unionOfSets(set4).printSet();
    cout << "Set 1 intersection Set 4: ";
    set1.intersectionOfSets(set4).printSet();

    cout << "Set 2 union Set 1: ";
    set2.unionOfSets(set1).printSet();
    cout << "Set 2 intersection Set 1: ";
    set2.intersectionOfSets(set1).printSet();

    cout << "Set 2 union Set 5: ";
    set2.unionOfSets(set5).printSet();
    cout << "Set 2 intersection Set 5: ";
```

```
    set2.intersectionOfSets(set5).printSet();

    cout << "Set 3 equal to Set 4: ";
    if (set3.isEqualTo(set4))
    {
        cout << "true" << endl;
    }
    else
    {
        cout << "false" << endl;
    }

    cout << "Set 3 equal to Set 5: ";
    if (set3.isEqualTo(set5))
    {
        cout << "true" << endl;
    }
    else
    {
        cout << "false" << endl;
    }

    cout << "Set 4 equal to Set 5: " ;
    if (set4.isEqualTo(set5))
    {
        cout << "true" << endl;
    }
    else
    {
        cout << "false" << endl;
    }

    return 0;
}
```

Run program & result:

```
cpp -o 2 } ; if ($?) { .\2 }
Set 1: 10 12 18 29
Set 2: 1 2 3 4 5
Set 2 after deletion: 1 2 4
Set 3: 8 10 12 14 15 18 19 20 21 29
Set 4: 8 10 12 14 15 18 19 20 21 29
Set 5: 3 7 8 10 12
Set 1 union Set 4: 8 10 12 14 15 18 19 20 21 29
Set 1 intersection Set 4: 10 12 18 29
Set 2 union Set 1: 1 2 4 10 12 18 29
Set 2 intersection Set 1: None
Set 2 union Set 5: 1 2 3 4 7 8 10 12
Set 2 intersection Set 5: None
Set 3 equal to Set 4: true
Set 3 equal to Set 5: false
Set 4 equal to Set 5: false
PS D:\VS CODE\C C++\CS360\HW#3>
```