using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CardClassLibrary

{

public class Hand

{

private List<Card> cards = new List<Card>();

public Hand() { }

public int Count()

{

int i = 0;

foreach (Card count in cards)

{

i++;

}

return i;

}

public int NumCards

{

get

{

return cards.Count;

}

}

public Card GetCard(int index)

{

return cards[index];

}

public int IndexOf(Card newCard)

{

return cards.IndexOf(newCard);

}

public int IndexOf(int value)

{

for(int i = 0; i < NumCards; i ++)

{

if (cards[i].Value == value) // i is index of card in the cards list. Value is the property of the card.

return i;

}

return -1; // return if out of scope

}

public int IndexOf(int value, int suit)

{

for (int i = 0; i < NumCards; i++)

{

if (cards[i].Value == value && cards[i].Suit == suit) // i is index of a card in cards list. Value is the property of

{

return i;

}

}

return -1; // return if out of scope \*/

}

/\*public int IndexOf(int value, int suit)

{

Card c = new Card(value, suit);

return IndexOf(c);

}\*/

public bool HasCard(Card searchCard)

{

foreach (Card cardInHand in cards)

{

if (searchCard.Value == cardInHand.Value && searchCard.Suit == cardInHand.Suit)

return true;

}

return false;

}

public bool HasCard(int searchValue)

{

foreach (Card cardInHand in cards)

{

if (searchValue == cardInHand.Value)

return true;

}

return false;

}

public bool HasCard(int searchValue, int searchSuit)

{

foreach (Card c in cards)

{

if (searchValue == c.Value && searchSuit == c.Suit)

return true;

}

return false;

}

public void Add(Card card)

{

cards.Add(card);

}

public Card Discard(int index)

{

Card c1 = cards[index];

cards.RemoveAt(index);

return c1;

}

public override string ToString()

{

string output = "";

foreach (Card c in cards)

{

output += c.ToString() + "\n";

}

return output;

}

}

}