**Project 6 – Cards**

**(20 points)**

**Name:** Ozaner Hansha

**Due Date: Tuesday, December 22, 2015**

**Description:**

In this project, we will start building some of the foundation for card games. We have been learning about enumerations and interfaces, in addition to our developing GUI and event-driven programming skills. In this project, you will get to try out several of these ideas. Be sure to include Javadoc and write clean, commented, nicely formatted code.

Carry out the following steps:

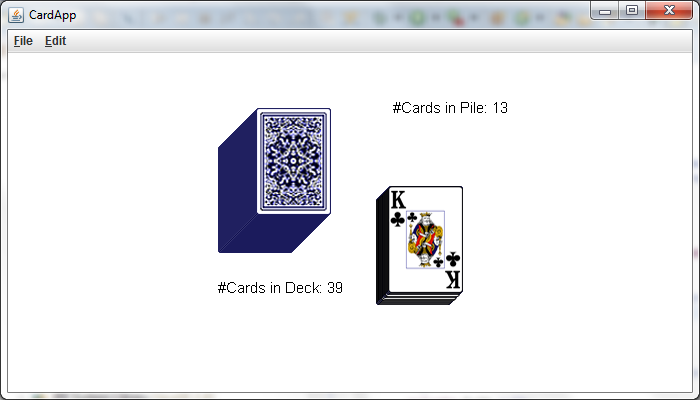
1. Download the archived project APCS2015Proj06Cards.
2. Create a Suit enumeration with the suits ***CLUBS***, ***DIAMONDS***, ***HEARTS***, ***SPADES***.  
     
   Ensure that the toString() method for Suit, produces the lowercase strings “clubs”, “diamonds”, “hearts”, and “spades”.
3. Create a Rank enumeration with the ranks ***DEUCE, THREE, FOUR, FIVE, SIX, SEVEN, EIGHT, NINE, TEN, JACK, QUEEN, KING, ACE***.   
     
   Ensure that the toString() method for Rank, produces the strings “2”, “3”, “4”, “5”, “6”, “7”, “8”, “9”, “10”, “j”, “q”, “k”, and ”a”. (This will turn out to be convenient when you construct filenames for the image files.)
4. Complete the GCard class, a subclass of GCompound, which implements the Card interface (which I have provided for you). Besides the methods required by the Card interface, there are some static convenience methods for you to implement as well.  
     
   Note that the project already contains a folder images with playing card images of the front and back of the cards.
5. Complete the Deck class, which is a subclass of ArrayList<Card>, since a deck is essentially just a list of cards.
6. Complete the Hand class, which is also a subclass of ArrayList<Card>. We won’t be using Hand in this project, but the concept occurs in many card games.
7. Complete the CardApp class, which is a subclass of GraphicsProgram.

In the init method of CardApp, make a deck of cards and place them face down on the canvas (in a pile) slightly left of center.  
  
On a mouse click over the deck, remove the top card (if one exists), flip it over face up and place it in a discard pile slightly right of center.  
  
On a mouse click over the discard pile (if one exists), return the top card, flip it over face down and return in to the deck pile.  
  
Ignore any other mouse clicks that are not over the deck or the discard pile.

To complete the project,

* submit this Word document to your shared Google Drive folder
* fill in your name above
* include the source listings for all of your classes (except the interface Card which I created)
* a screen shot of the app after a few cards have been dealt

***Screenshot***



***CardApp***

**package** unit5.cardGame;

**import** java.awt.Dimension;

**import** java.awt.Font;

**import** java.awt.event.MouseEvent;

**import** acm.graphics.GLabel;

**import** acm.program.GraphicsProgram;

/\*\*

\* This program places a deck of cards on the screen and when

\* clicked moves the top card to a discard pile. This process can be

\* reversed by clicking on the discard pile.

\* <br><br>

\* Project 6 - CardApp<br>

\* AP Computer Science - Pd. 7<br>

\* Dr. Jones

\* **@author** Ozaner Hansha

\*/

@SuppressWarnings("serial")

**public** **class** CardApp **extends** GraphicsProgram

{

//Window and size constants.

**private** **static** **final** Dimension ***INITIAL\_DIM*** = **new** Dimension(700,400);

**private** **static** **final** **double** ***APP\_WIDTH\_3X*** = ***INITIAL\_DIM***.getWidth()\*.3;

**private** **static** **final** **double** ***APP\_WIDTH\_6X*** = ***INITIAL\_DIM***.getWidth()\*.6;

**private** **static** **final** **double** ***CARD\_Y*** = ***INITIAL\_DIM***.getHeight()\*.5 - GCard.*cardHeight*();

//Deck variables

**private** **static** Deck *deck* = GCard.*makeDeck*();

**private** **static** **int** *DECK\_HEIGHT*; //Keeps track of height of deck on canvas.

//Deck Counter Variables

**private** **static** **final** String ***DECK\_COUNT\_LABEL\_STRING*** = "#Cards in Deck: ";

**private** **static** GLabel *DECK\_COUNT\_LABEL* = **new** GLabel(***DECK\_COUNT\_LABEL\_STRING*** + *deck*.size());

**private** **static** **final** **double** ***DECK\_COUNTER\_X*** = ***INITIAL\_DIM***.getWidth()\*.3;

**private** **static** **final** **double** ***DECK\_COUNTER\_Y*** = ***INITIAL\_DIM***.getHeight()\*.6;

//Pile Counter Variables

**private** **static** **final** String ***PILE\_COUNT\_LABEL\_STRING*** = "#Cards in Pile: ";

**private** **static** GLabel *PILE\_COUNT\_LABEL* = **new** GLabel(***PILE\_COUNT\_LABEL\_STRING*** + 0);

**private** **static** **final** **double** ***PILE\_COUNTER\_X*** = ***INITIAL\_DIM***.getWidth()\*.55;

**private** **static** **final** **double** ***PILE\_COUNTER\_Y*** = ***INITIAL\_DIM***.getHeight()\*.15;

**private** **static** **final** Font ***COUNTER\_FONT*** = **new** Font("Ariel", 0, 15);

/\*\*

\* Starts the CardApp.

\* **@param** args - Not expecting any command arguments.

\*/

**public** **static** **void** main(String[] args)

{

**new** CardApp().start(args);

}

/\*\*

\* Make a deck of cards and place them face down on the canvas (in a pile) slightly left of center.

\*/

@Override

**public** **void** init()

{

setSize(***INITIAL\_DIM***);

addMouseListeners();

*deck*.shuffle(); //Shuffles the deck

//Adds all cards in deck to canvas.

**for**(Card c: *deck*)

{

add((GCard)c, ***APP\_WIDTH\_3X*** + *DECK\_HEIGHT*, ***CARD\_Y*** - *DECK\_HEIGHT*);

*DECK\_HEIGHT*++;

}

*DECK\_COUNT\_LABEL*.setFont(***COUNTER\_FONT***);

*PILE\_COUNT\_LABEL*.setFont(***COUNTER\_FONT***);

add(*DECK\_COUNT\_LABEL*,***DECK\_COUNTER\_X***,***DECK\_COUNTER\_Y***); //Adds deck counter to canvas

add(*PILE\_COUNT\_LABEL*,***PILE\_COUNTER\_X***,***PILE\_COUNTER\_Y***); //Adds pile counter to canvas

}

/\*\*

\* On a mouse click over the deck, remove the top card (if one exists), flip it over face up and place

\* it in a discard pile slightly right of center.

\* On a mouse click over the discard pile (if one exists), return the top card, flip it over face down

\* and return in to the deck pile.

\* Ignore any other mouse clicks that are not over the deck or the discard pile.

\*/

@Override

**public** **void** mouseClicked(MouseEvent e)

{

//Sets obj equal to the top card in the deck/pile

GCard obj = (GCard)getElementAt(e.getX(),e.getY());

**if**(obj != **null**) //Checks to see if a card was clicked.

{

**if**(*DECK\_HEIGHT* > *deck*.indexOf(obj)) //If this card is on the deck.

{

obj = (GCard)*deck*.get(*DECK\_HEIGHT*-1); //The top card of the deck

obj.setLocation(***APP\_WIDTH\_6X*** - *DECK\_HEIGHT*, ***CARD\_Y*** + *DECK\_HEIGHT*); //Right side of screen.

*DECK\_HEIGHT*--;

}

**else** //Card must be on pile

{

obj = (GCard)*deck*.get(*DECK\_HEIGHT*); //The top card of the pile

obj.setLocation(***APP\_WIDTH\_3X*** + *DECK\_HEIGHT*, ***CARD\_Y*** - *DECK\_HEIGHT*); //Left side of screen.

*DECK\_HEIGHT*++;

}

obj.flipOver();

obj.sendToFront();

*DECK\_COUNT\_LABEL*.setLabel(***DECK\_COUNT\_LABEL\_STRING*** + *DECK\_HEIGHT*);

*PILE\_COUNT\_LABEL*.setLabel(***PILE\_COUNT\_LABEL\_STRING*** + (*deck*.size() - *DECK\_HEIGHT*));

}

}

}

***GCard***

**package** unit5.cardGame;

**import** acm.graphics.GCompound;

**import** acm.graphics.GImage;

/\*\*

\* Models a (graphical) playing card. This model includes not only the

\* rank and suit, but also the graphical representation of the card.

\*

\* **@author** Dr. Mark A. Jones

\*/

@SuppressWarnings("serial")

**public** **class** GCard **extends** GCompound **implements** Card

{

/\*\*

\* Rank of the card. Initialized in constructor.

\*/

**private** Rank rank;

/\*\*

\* Suit of the card. Initialized in constructor.

\*/

**private** Suit suit;

/\*\*

\* Image on front of card. Initialized in constructor.

\*/

**public** GImage faceUpImage;

/\*\*

\* Face down image of all cards.

\*/

**private** **static** **final** GImage ***FACE\_DOWN\_IMAGE*** = **new** GImage("back-blue-75-1.png");

/\*\*

\* Create a playing card with a given rank and suit.

\* **@param** rank - the rank

\* **@param** suit - the suit

\*/

**public** GCard(Rank rank, Suit suit)

{

**this**.rank = rank;

**this**.suit = suit;

faceUpImage = **new** GImage(String.*format*("%s-%s-75.png", suit.toString(), rank.toString()));

add(**new** GImage(***FACE\_DOWN\_IMAGE***.getImage())); //This is below.

add(faceUpImage); //This is on top.

turnFaceDown(); //Starts face down.

}

/\*\*

\* Get the rank of the card.

\* **@return** the rank

\*/

@Override

**public** Rank getRank()

{

**return** rank;

}

/\*\*

\* Get the suit of the card.

\* **@return** the suit

\*/

@Override

**public** Suit getSuit()

{

**return** suit;

}

/\*\*

\* A card as a string.

\* **@return** the card as a string

\*/

@Override

**public** String toString()

{

**return** String.*format*("Rank: %s, Suit: %s", getRank(), getSuit());

}

/\*\*

\* Is the card face up?

\* **@return** true if the card is face up

\*/

@Override

**public** **boolean** isFaceUp()

{

**return** faceUpImage.isVisible();

}

/\*\*

\* Turn the card face up.

\* **@return** the card

\*/

@Override

**public** GCard turnFaceUp()

{

faceUpImage.setVisible(**true**);

**return** **this**;

}

/\*\*

\* Turn the card face down.

\* **@return** the card

\*/

@Override

**public** GCard turnFaceDown()

{

faceUpImage.setVisible(**false**);

**return** **this**;

}

/\*\*

\* Flip the card over.

\* **@return** the card

\*/

@Override

**public** GCard flipOver()

{

faceUpImage.setVisible(!faceUpImage.isVisible());

**return** **this**;

}

/\*\*

\* Convenience method for making a deck of GCards.

\* **@return** the new deck

\*/

**public** **static** Deck makeDeck()

{

Card[] cards = **new** Card[52];

**int** i = 0;

**for**(Suit s: Suit.*values*())

{

**for**(Rank r: Rank.*values*())

{

cards[i] = **new** GCard(r,s);

i++;

}

}

**return** **new** Deck(cards);

}

/\*\*

\* Get the width for any playing card.

\* **@return** the width

\*/

**public** **static** **double** cardWidth()

{

**return** ***FACE\_DOWN\_IMAGE***.getWidth();

}

/\*\*

\* Get the height for any playing card.

\* **@return** the height

\*/

**public** **static** **double** cardHeight()

{

**return** ***FACE\_DOWN\_IMAGE***.getHeight();

}

/\*\*

\* Get the back image for any playing card.

\* **@return** the back image

\*/

**public** **static** GImage getBackImage()

{

**return** ***FACE\_DOWN\_IMAGE***;

}

}

***Deck***

**package** unit5.cardGame;

**import** java.util.ArrayList;

**import** java.util.Collections;

/\*\*

\* Models a standard deck of playing cards.

\* **@author** Dr. Mark A. Jones

\*/

@SuppressWarnings("serial")

**public** **class** Deck **extends** ArrayList<Card>

{

/\*\*

\* Make a new deck of an arbitrary amount cards.

\*/

**public** Deck(Card... cards)

{

**for**(Card c: cards)

add(c);

}

/\*\*

\* Deal a card.

\* **@return** the top card from the deck, or null if empty.

\*/

**public** Card deal()

{

**return** size() != 0 ? remove(size()-1) : **null**;

}

/\*\*

\* Shuffle the remaining cards in the deck.

\*/

**public** **void** shuffle()

{

Collections.*shuffle*(**this**);

}

}

***Hand***

**package** unit5.cardGame;

**import** java.util.ArrayList;

/\*\*

\* Models a hand of an arbitrary amount of {@link GCards}.

\* **@author** Ozaner Hansha

\*/

@SuppressWarnings("serial")

**public** **class** Hand **extends** ArrayList<Card>

{

/\*\*

\* Make a new hand of an arbitrary amount cards.

\*/

**public** Hand(Card... cards)

{

**for**(Card c: cards)

add(c);

}

}

***Rank***

**package** unit5.cardGame;

/\*\*

\* The constants corresponding the {@link Card} ranks.

\* **@author** Ozaner Hansha

\*/

**public** **enum** Rank

{

***DEUCE***("2"),***THREE***("3"),***FOUR***("4"),***FIVE***("5"),***SIX***("6"),***SEVEN***("7"),***EIGHT***("8"),

***NINE***("9"),***TEN***("10"),***JACK***("j"),***QUEEN***("q"),***KING***("k"),***ACE***("a");

/\*\*

\* The rank of this card.

\*/

**private** String rank;

/\*\*

\* Constructs a new card with a given rank.

\* **@param** r - The rank.

\*/

**private** Rank(String r)

{

rank = r;

}

/\*\*

\* Returns the {@link #rank} of this card.

\*/

**public** String toString()

{

**return** rank;

}

}

***Suit***

**package** unit5.cardGame;

/\*\*

\* The constants corresponding to {@link Card} suits.

\* **@author** Ozaner Hansha

\*/

**public** **enum** Suit

{

***CLUBS***,***DIAMONDS***,***HEARTS***,***SPADES***;

/\*\*

\* Returns the name of the suit in lowercase.

\*/

@Override

**public** String toString()

{

**return** name().toLowerCase();

}

}