//card.java

/\*\*

\* SYST 17796 Project Winter 2019 Base code.

\* Students can modify and extend to implement their game.

\* Add your name as a modifier and the date!

\*/

package ca.sheridancollege.project;

/\*\*

\* A class to be used as the base Card class for the project. Must be general

\* enough to be instantiated for any Card game. Students wishing to add to the

\* code should remember to add themselves as a modifier.

\*

\* @author dancye, 2018

\* @modifier data pirates

\*/

public abstract class Card {

public enum Colors {

Red, Yellow, Green, Blue

};

public enum Ranks {

ZERO, ONE, TWO, THREE, FOUR, FIVE, SIX, SEVEN, EIGHT, NINE, SKIP, REVERSE, DRAWTWO, DRAWFOUR, WILDCARD

};

private final Colors colour;

private final Ranks rank;

//default modifier for child classes

/\*\*

\* Students should implement this method for their specific children classes

\*

\* @return a String representation of a card. Could be an UNO card, a

\* regular playing card etc.

\*/

/\*

We have taken the reference from the inclass assignment of uno game.

Source Inclass Assigement ICE02

\*/

public Card(Colors c, Ranks r) {

colour = c;

rank = r;

}

public Ranks getValue() {

return this.rank;

}

public Colors getColors() {

return this.colour;

}

@Override

public abstract String toString();

}

//default modifier for child classes

/\*\*

\* Students should implement this method for their specific children classes

\*

\* @return a String representation of a card. Could be an UNO card, a regular

\* playing card etc.

\*/

//groupofcard

public class GroupOfCards {

//The group of cards, stored in an ArrayList

private ArrayList<Card> cards;

private int size;//the size of the grouping

public GroupOfCards(int givenSize) {

size = givenSize;

}

/\*\*

\* A method that will get the group of cards as an ArrayList

\*

\* @return the group of cards.

\*/

public ArrayList<Card> showCards() {

return cards;

}

public void shuffle() {

Collections.shuffle(cards);

}

/\*\*

\* @return the size of the group of cards

\*/

public int getSize() {

return size;

}

/\*\*

\* @param givenSize the max size for the group of cards

\*/

public void setSize(int givenSize) {

size = givenSize;

}

}//end class

//game.java

/\*\*

\* SYST 17796 Project Winter 2019 Base code.

\* Students can modify and extend to implement their game.

\* Add your name as a modifier and the date!

\*/

package ca.sheridancollege.project;

import java.util.ArrayList;

/\*\*

\* The class that models your game. You should create a more specific

\* child of this class and instantiate the methods given.

\* @author dancye, 2018

\* @modifier data pirates

\*/

public abstract class Game

{

private int handSize = 60;

public Card[] cards = new Card[handSize];

public void generateHand()

{

int countCards = 0;

for(Card.Colors s: Card.Colors.values())

{

for(Card.Ranks v: Card.Ranks.values())

{

cards[countCards] = (new Card(s,v) {

@Override

public String toString() {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

});

countCards++;

}

}//end outter for

}

private final String gameName;//the title of the game

private ArrayList <Player> players;// the players of the game

public Game(String givenName)

{

gameName = givenName;

players = new ArrayList();

}

/\*\*

\* @return the gameName

\*/

public String getGameName()

{

return gameName;

}

/\*\*

\* @return the players of this game

\*/

public ArrayList <Player> getPlayers()

{

return players;

}

/\*\*

\* @param players the players of this game

\*/

public void setPlayers(ArrayList <Player> players)

{

this.players = players;

}

/\*\*

\* Play the game. This might be one method or many method calls depending

\* on your game.

\*/

public abstract void play();

/\*\*

\* When the game is over, use this method to declare and display a winning

\* player.

\*/

public abstract void declareWinner();

}//end class

//player.java

/\*\*

\* SYST 17796 Project Winter 2019 Base code.

\* Students can modify and extend to implement their game.

\* Add your name as a modifier and the date!

\*/

package ca.sheridancollege.project;

/\*\*

\* A class that models each Player in the game. Players have an identifier, which should be unique.

\* @author dancye, 2018

\* @modifier data pirates

\*/

public abstract class Player

{

private String playerID; //the unique ID for this player

/\*\*

\* A constructor that allows you to set the player's unique ID

\* @param name the unique ID to assign to this player.

\*/

public Player(String name)

{

playerID= name;

}

/\*\*

\* @return the playerID

\*/

public String getPlayerID()

{

return playerID;

}

/\*\*

\* Ensure that the playerID is unique

\* @param givenID the playerID to set

\*/

public static void main(String[] args){

Game ch = new Game() {

@Override

public void play() {

this.play(); //To change body of generated methods, choose Tools | Templates.

}

@Override

public void declareWinner() {

this.declareWinner();//To change body of generated methods, choose Tools | Templates.

}

};

ch.generateHand();

for(Card c: ch.cards)

{

System.out.println(c.getValue() + " of " + c.getColors());

}

}

public void setPlayerID(String givenID)

{

playerID = givenID;

}

/\*\*

\* The method to be instantiated when you subclass the Player class

\* with your specific type of Player and filled in with logic to play your game.

\*/

public abstract void play();

}