

UCCD2044 Object-Oriented Programming

OOP Programming Assignment

War Card Game

|  |  |  |
| --- | --- | --- |
| Student Name | Student ID | Course |
| Wan Kar Hou | 1607221 | CS |
| Lim Ming Yuen | 1803655 | CS |
| Darshan A/L Suresh | 1606423 | CS |
| Chang Wei Li | 1605052 | CN |

* **Program description**

In this assignment, we have been told to develop a card game using Java and Object-Oriented Programming technique. This card game is played using a standard 52-card deck which has (13 ranks of 4 suits).

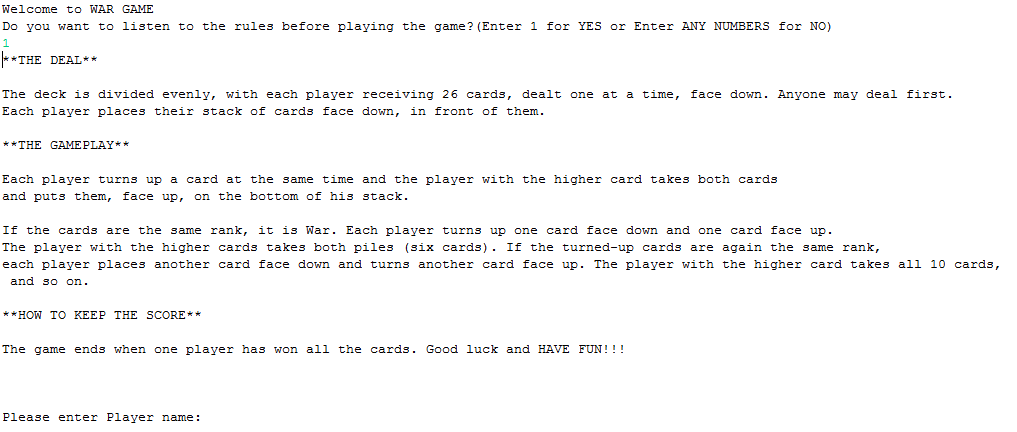
* **Game description**

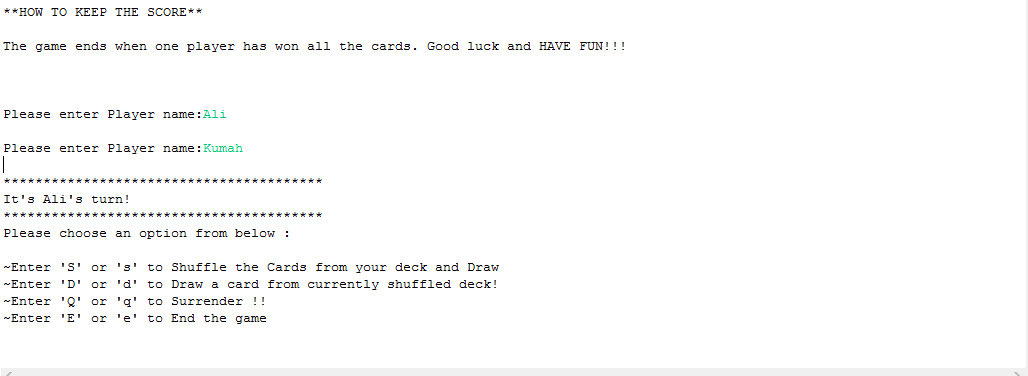
The deck is divided evenly, with each player receiving 26 cards, dealt one at a time, face down. Anyone may deal first. Each player places his stack of cards face down, in front of him.

Each player turns up a card at the same time and the player with the higher card takes both cards and puts them, face down, on the bottom of his stack. If the cards are the same rank, it is War, player saying "W - A - R" and each player turns up three card face down and one card face up. The player with the higher cards takes both piles (ten cards). If the turned-up cards are again the same rank, then return all the card to each player.

The game continues until one player has all the cards and wins. If you don't have enough cards to complete the war, you lose. If neither player has enough cards, the one who runs out first loses. If both run out simultaneously, it's a draw.

* **How does the program work and play**

****When the program initials, it will show the name of the game which is “War Game”, and some basic rule to assist the player how to score and win the game.



Picture 1

First of all, based on the above picture 1 the players have to insert their nickname to start the game. After that, there is four option for player to choose which is ‘S/s’, ‘D/d’, ‘Q/q’ or ‘E/e’.

Option 1

‘S/s’ If the player enters ‘S/s’ is to shuffle the cards from the deck and draw one card.

Option 2

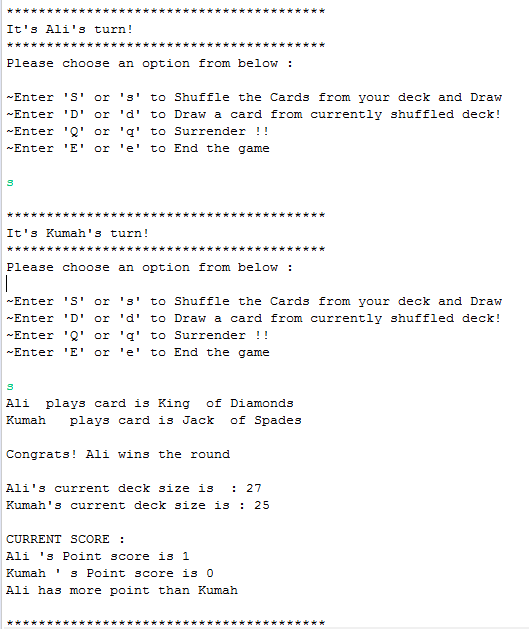
‘D/s’ If the player enters ‘D/d’ is to draw one card from currently shuffled deck.

Option 3

‘Q/q’ If the player enters ‘Q/q’ is to surrender.

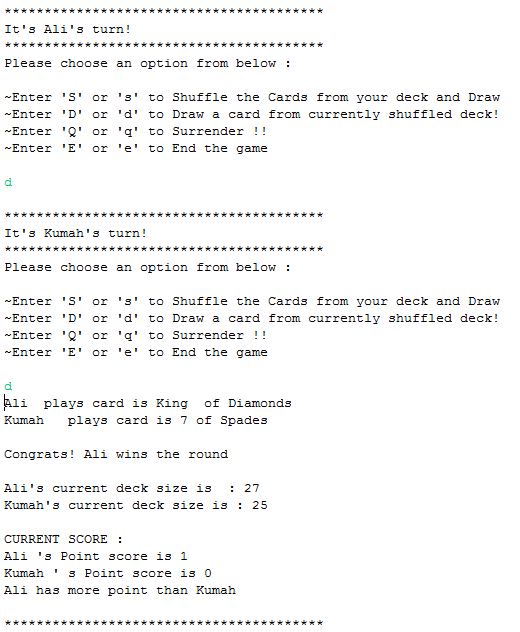
Option 4

‘E/e’ If the player enters ‘E/e’ is to end the game.



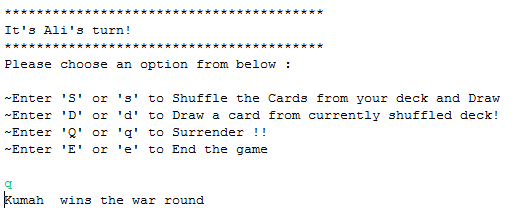
Picture 2

Picture 2 is an example of option 1 ‘S/s’ which is for each player to shuffle the cards from the deck and draw one card. After that compare the rate between two cards.



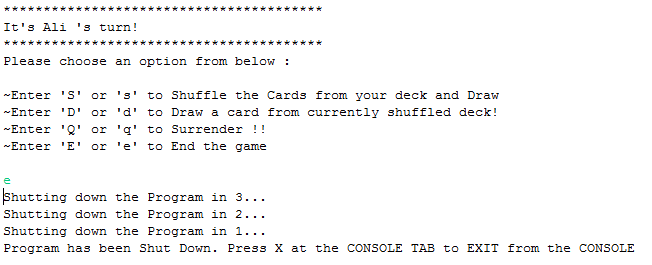
Picture 3

Picture 3 is an example of option 2 ‘D/d’ which is for each player to draw one card from currently shuffled deck. After that compare the rate between two cards.



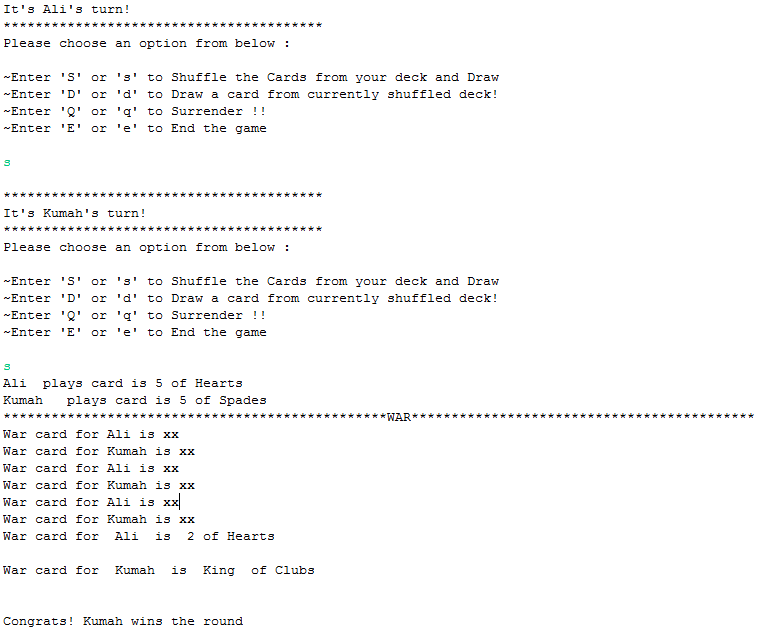
Picture 4

Picture 4 is an example of option 3 ‘Q/q’ which is for player to surrender. If player 1 surrender then player 2 will automatically win the game.



Picture 5

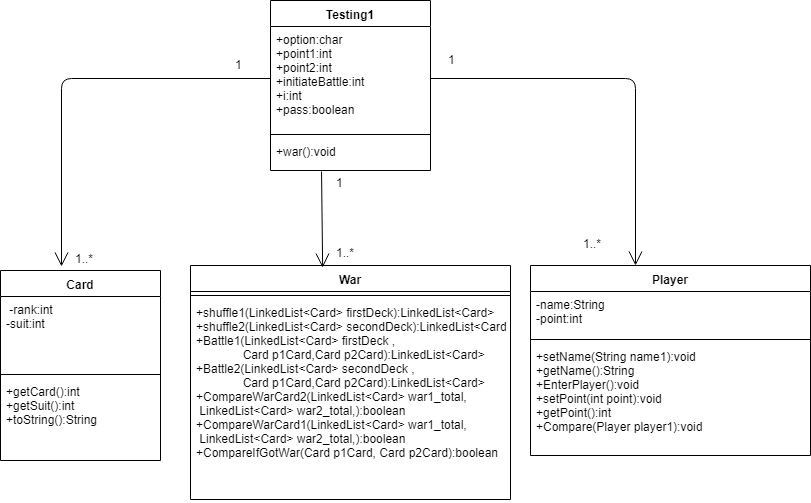
Picture 5 is an example of option 4 ‘E/e’ which is for player to end/terminated the game.



Picture 6

Picture 6 is an example when war is happening. Base on the above situation Ali and Kumah draw the same rate of card which is 5 so WAR happen, but at last Kumah wins the round so Kumah is able to receive 10 cards.

UML Class Diagram



Java source code(Player class)

import java.util.Scanner;

public class Player {

private String name;

private int point;

public void setName(String name1) { // set the Name of the Player

name=name1;

}

public String getName() { //Get the name of the player

return name;

}

public void EnterPlayer() //Enter the name of the player

{ getName();

Scanner obj1 = new Scanner(System.in) ;

System.out.print("\nPlease enter Player name:");

name = obj1.nextLine() ;

}

public void setPoint(int point) //Set the point score by the player

{

this.point = point ;

}

public int getPoint() //Get the point score by the player

{

return point;

}

public void Compare(Player player1 )//Compare the score point of each player and declare who had more point

{

if(player1.getPoint()> this.getPoint())

{

System.out.println(player1.getName() +" has more point than " +this.getName()) ;

}

else if(player1.getPoint()<this.getPoint())

{

System.out.println(this.getName() +" has more point than " +player1.getName()) ;

}

else

{

System.out.println("Both players point are same currently");

}

}

}

Java source code (Card class)

class Card {// Card class are immutable

private int rank; //initialize the rank (Ace,2,3,4...King)

private int suit; //initialize the suit (spades, hearts...)

//constructor

public Card(int suit, int rank){

this.rank = rank;

this.suit = suit;

}//end construcor

//getter method

public int getCard(){

return rank;

}//end getCard

//setter method

public int getSuit() //Get the suit of the card

{

return suit;

}

@Override

public String toString(){

//combine rank and suit together into a single string(ex: Ace of Diamonds

//suing StringBuilhowCardder for modifiability later on

StringBuilder displayCard = new StringBuilder();

//create if looping for card rank

if(rank == 11) { displayCard.append("Jack "); }

else if (rank == 12) { displayCard.append("Queen "); }

else if (rank == 13) { displayCard.append("King "); }

else if (rank == 1) { displayCard.append("Ace "); }

else { displayCard.append(rank);}

//end if looping for card rank

displayCard.append(" of "); //Print the string format

//create if looping for card suit

if(suit == 0) { displayCard.append("Spades "); }

else if (suit == 1) { displayCard.append("Hearts "); }

else if (suit == 2) { displayCard.append("Clubs "); }

else if (suit == 3) { displayCard.append("Diamonds "); }

else { }

//end if looping for card suit

//return the result of an entire combined string

return displayCard.toString();

}

}//end Card Class

Java source code (War class)

import java.util.ArrayList;

import java.util.Collections;

import java.util.LinkedList;

import java.util.List;

import java.util.Random;

import java.util.Scanner;

import javafx.util.Pair;

public class War{

public LinkedList<Card> shuffle1(LinkedList<Card> firstDeck) //Shuffle player 1 deck randomly

{

Collections.shuffle(firstDeck, new Random());

return firstDeck;

}

public LinkedList<Card >shuffle2(LinkedList<Card> secondDeck) //Shuffle player 2 deck randomly

{

Collections.shuffle(secondDeck, new Random());

return secondDeck;

}

public LinkedList<Card> Battle1(LinkedList<Card> firstDeck, Card p1Card,Card p2Card) //Add Card to player 1 deck if player1 won the round

{

firstDeck.addLast(p1Card);

firstDeck.addLast(p2Card);

return firstDeck ;

}//end if

public LinkedList<Card> Battle2(LinkedList<Card> secondDeck,Card p2Card,Card p1Card) { //Add card to player 2 deck after player 2 won the round

secondDeck.addLast(p2Card);

secondDeck.addLast(p1Card);

return secondDeck;

}

public boolean CompareWarCard2(LinkedList<Card> war1\_total ,LinkedList<Card> war2\_total)//Declare Player2 win if 5th card in war is bigger than Player 1 5th card

{

if(war2\_total.get(4).getCard() >war1\_total.get(4).getCard())

{

return true;

}

else

{

return false;

}

}

public boolean CompareWarCard1(LinkedList<Card> war1\_total, LinkedList<Card> war2\_total) //Declare Player1 win if 5th card in war is bigger than Player 2 5th card

{

if(war1\_total.get(4).getCard() >war2\_total.get(4).getCard())

{

return true;

}

else

{

return false;

}

}

public boolean CompareIfGotWar(Card p1Card, Card p2Card)//Compare and Declare W-A-R if both cards are in same rank

{

if(p1Card.getCard() == p2Card.getCard())

{

return true;

}

else

{

return false;

}

}

}

Java source code (Testing1 class)

import java.util.ArrayList; //import ArrayList

import java.util.Random; //import Random

import java.util.Scanner;

import javax.swing.JFrame;

import javax.swing.JOptionPane;

import java.util.List; //import List

import java.util.Collections; //import Collections

import java.util.InputMismatchException;

import java.util.LinkedList; //import LinkList

public class Testing1 {

public static void main(String[] args) {

char option;

int point1=0; //Initialize Score Point for Player 1

int point2=0; //Initialize Score Point for Player 2

int initiateBattle =0;

int i =0 ;

Player p1=new Player(); //Initialize Player 1

Player p2=new Player(); //Initialize Player 2

boolean pass ;

List<Card> DeckOfCard = new ArrayList<Card>(); //create an ArrayList "DeckOfCard"

//initialize War part

War War1 = new War() ; // For Player1

War War2 = new War() ; //For Player 2

War Compare = new War();

for(int x=0; x<4; x++){ //0-3 for suit (4 suits)

for(int y=1; y<14; y++){ //1-13 for rank (13 ranks)

DeckOfCard.add(new Card(x,y)); //create new card and add into the deckOfCard

} //end rank for

}//end suit for

Collections.shuffle(DeckOfCard, new Random()); //shuffle the deck randomly

//creating 2 decks, each for player1/player2

LinkedList<Card> firstDeck = new LinkedList<Card>();

LinkedList<Card> secondDeck = new LinkedList<Card>();

//Cards from DeckOfCard are divided into firstDeck and Second Deck

firstDeck.addAll(DeckOfCard.subList(0, 26)); //26 cards of firstDeck

secondDeck.addAll(DeckOfCard.subList(26,DeckOfCard.size()));//26 cards of SecondDeck

// Display the main menu of the Card Game

System.out.print("Welcome to WAR GAME\n") ;

System.out.println("Do you want to listen to the rules before playing the game?(Enter 1 for YES or Enter ANY NUMBERS for NO)");

try

{Scanner scn = new Scanner(System.in);

int input = scn.nextInt();

if((input==1))

{

System.out.println("\*\*THE DEAL\*\*\r\n" +

"\r\n" +

"The deck is divided evenly, with each player receiving 26 cards, dealt one at a time, face down. Anyone may deal first."

+ "\nEach player places their stack of cards face down, in front of them.\r\n" +

"\n\*\*THE GAMEPLAY\*\*\r\n" +

"\r\n" +

"Each player turns up a card at the same time and the player with the higher card takes both cards "

+ "\nand puts them, face up, on the bottom of his stack.\r\n" +

"\r\n" +

"If the cards are the same rank, it is War. Each player turns up one card face down and one card face up."

+ "\nThe player with the higher cards takes both piles (six cards). If the turned-up cards are again the same rank, "

+ "\neach player places another card face down and turns another card face up. The player with the higher card takes all 10 cards,"

+ "\n and so on.\r\n" +

"\n\*\*HOW TO KEEP THE SCORE\*\*\r\n" +

"\r\n" +

"The game ends when one player has won all the cards. Good luck and HAVE FUN!!! \n\n");

}

}

catch(InputMismatchException e)

{

System.out.println("Please ONLY ENTER NUMBERS!\n");

main(args);

}

p1.EnterPlayer(); //Enter Player 1 name

p2.EnterPlayer(); //Enter Player 2 name

while(true){

initiateBattle =0;

pass = true ;

while(pass == true)

{

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("It's " + p1.getName() + "'s turn!");

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("Please choose an option from below : \n" ) ;

System.out.println("~Enter 'S' or 's' to Shuffle the Cards from your deck and Draw");

System.out.println("~Enter 'D' or 'd' to Draw a card from currently shuffled deck! ");

System.out.println("~Enter 'Q' or 'q' to Surrender !! ");

System.out.println("~Enter 'E' or 'e' to End the game\n");

Scanner inputOption = new Scanner(System.in) ;

option = inputOption.next().trim().charAt(0);

if(option == 'S' || option == 's')

{

War1.shuffle1(firstDeck);

initiateBattle += 1 ;

pass = false ;

}

else if(option == 'D' || option == 'd')

{

initiateBattle+= 1 ;

pass = false ;

}

else if(option == 'Q' || option == 'q')

{

System.out.println(p2.getName()+ " " + " wins the war round\n");

main(args) ;

}

else if(option == 'E' || option == 'e')

{

System.exit(0);

}

else

{

System.out.print("Invalid Input");

System.out.print("==>Please enter only the given Options!<==\n\n");

i++ ;

if(i==2)

{

main(args) ;

}

else

{

pass= true ;

}

}

}

i =0;

pass = true ;

while(pass == true)

{

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("It's " + p2.getName() + "'s turn!");

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("Please choose an option from below : \n" ) ;

System.out.println("~Enter 'S' or 's' to Shuffle the Cards from your deck and Draw");

System.out.println("~Enter 'D' or 'd' to Draw a card from currently shuffled deck! ");

System.out.println("~Enter 'Q' or 'q' to Surrender !! ");

System.out.println("~Enter 'E' or 'e' to End the game\n");

Scanner inputOption = new Scanner(System.in) ;

option = inputOption.next().trim().charAt(0);

if(option == 'S' || option == 's')

{

War2.shuffle2(secondDeck);

initiateBattle += 1 ;

pass = false ;

}

else if(option == 'D' || option == 'd')

{

initiateBattle+= 1 ;

pass = false ;

}

else if(option == 'Q' || option == 'q')

{

System.out.println(p1.getName()+ " " + " wins the war round\n");

main(args) ;

}

else if(option == 'E' || option == 'e')

{

System.exit(0);

}

else

{

System.out.print("Invalid Input");

System.out.print("==>Please enter only the given Options!<==\n\n");

i++ ;

if(i==2)

{

main(args);

}

else

{

pass= true ;

}

}

i=0;

}

//rank comparation between two cards

// When initiateBattle==2 , means both player agree to continue the game

if(initiateBattle ==2 )

{

Card p1Card = firstDeck.pop(); //each player place one card face up

Card p2Card = secondDeck.pop();

System.out.print(p1.getName() +" "+" plays card is " + p1Card.toString()+"\n");

System.out.print(p2.getName()+" "+" plays card is " + p2Card.toString()+"\n");

//Initialize the War Part

if(p1Card.getCard() > p2Card.getCard()){

firstDeck = War1.Battle1(firstDeck, p2Card, p1Card);

System.out.println("\nCongrats! " + p1.getName() +" wins the round\n");

System.out.println(p1.getName() + "'s current deck size is : " + firstDeck.size());

System.out.println(p2.getName() + "'s current deck size is : " + secondDeck.size());

point1++;

p1.setPoint(point1);

System.out.println("\nCURRENT SCORE : ");

System.out.println(p1.getName()+" 's Point score is "+ p1.getPoint()) ;

System.out.println(p2.getName()+" ' s Point score is "+ p2.getPoint()) ;

p1.Compare(p2);

}

else if(Compare.CompareIfGotWar(p1Card, p2Card)){ ////when nobody wins - it's a WAR! and both card are in same rank

war() ;

LinkedList<Card> war1 = new LinkedList<Card>();

LinkedList<Card> war2 = new LinkedList<Card>();

LinkedList<Card> war1\_total = new LinkedList<Card>() ;

LinkedList<Card> war2\_total = new LinkedList<Card>() ;

//Add the card into each Linked List to store

war1\_total.add(p1Card) ;

war2\_total.add(p2Card) ;

//creating war cards

//checking do players have enough (4)cards to stay in game

for(int x=0; x<3; x++){

//either one player runs out of card is game over

if((firstDeck.size() == 0)|| (secondDeck.size() == 0 )){

break;

}//end if

System.out.println("War card for "+p1.getName()+" is xx\nWar card for "+p2.getName()+" is xx");

war1.add(firstDeck.pop()); //place additional card for war

war2.add(secondDeck.pop());

}//end for

war1\_total.addAll(war1) ;//Add the war1 into war1\_total

war2\_total.addAll(war2) ;//Add the war2 into war2\_total

war1\_total.add(firstDeck.pop()) ;

war2\_total.add(secondDeck.pop()) ;

//only compare result when both players have enough cards for war

if(war1.size() == 3 && war2.size() == 3){

//display the war cards from each player

System.out.println("War card for "+" " +p1.getName()+" " +" is "+ " " +war1\_total.get(4).toString()+"\n");

System.out.println("War card for "+" "+p2.getName()+" " +" is " + " "+war2\_total.get(4).toString()+"\n");

//WHEN PLAYER 1 WIN THE WAR

if(Compare.CompareWarCard1(war1\_total, war2\_total)){

firstDeck.addAll(war1\_total); //player1 get all 10 cards

firstDeck.addAll(war2\_total);

firstDeck = War1.shuffle1(firstDeck);

System.out.println("\nCongrats! " + p1.getName() +" wins the round\n");

System.out.println(p1.getName() + "'s current deck size is : " + firstDeck.size());

System.out.println(p2.getName() + "'s current deck size is : " + secondDeck.size());

point1 = point1+5 ;

p1.setPoint(point1) ;

System.out.println("\n\nCURRENT SCORE : ");//Display current Score of both players

System.out.println(p1.getName()+" 's Point score is "+ p1.getPoint()) ;

System.out.println(p2.getName()+" ' s Point score is "+ p2.getPoint()) ;

p1.Compare(p2);

}//end if

//WHEN PLAYER 2 WINS THE WAR

else if(Compare.CompareWarCard2(war1\_total , war2\_total))

{

secondDeck.addAll(war1\_total); //player2 get all 10 cards

secondDeck.addAll(war2\_total);

secondDeck = War2.shuffle2(secondDeck);

System.out.println("\nCongrats! " + p2.getName() +" wins the round\n");

System.out.println(p1.getName() + "'s current deck size is : " + firstDeck.size());

System.out.println(p2.getName() + "'s current deck size is : " + secondDeck.size());

point2 = point2 +5 ;

p2.setPoint(point2);

System.out.println("\nCURRENT SCORE : ");

System.out.println(p1.getName()+" 's Point score is "+ p1.getPoint()) ;

System.out.println(p2.getName()+" ' s Point score is "+ p2.getPoint()) ;

p1.Compare(p2);

System.out.println(p2.getName()+" " + " wins the war round\n");

}

else

{

System.out.println("Its a draw for both sides\n");

System.out.println("Cards will be returned to each side\n");

secondDeck.addAll(war2\_total) ;

firstDeck.addAll(war1\_total);

firstDeck =War1.shuffle1(firstDeck);

secondDeck = War2.shuffle2(secondDeck);

System.out.println(p1.getName() + "'s current deck size is : " + firstDeck.size());

System.out.println(p2.getName() + "'s current deck size is : " + secondDeck.size());

System.out.println("\nCURRENT SCORE : ");

System.out.println(p1.getName()+" 's Point score is "+ p1.getPoint()) ;

System.out.println(p2.getName()+" ' s Point score is "+ p2.getPoint()) ;

}

}

}//end else if

else {//if player 2 win

secondDeck =War2.Battle2(secondDeck, p2Card, p1Card) ;

System.out.println("\nCongrats! " + p2.getName() +" wins the round\n");

System.out.println(p1.getName() + "'s current deck size is : " + firstDeck.size());

System.out.println(p2.getName() + "'s current deck size is : " + secondDeck.size());

point2++;

p2.setPoint(point2);

System.out.println("\nCURRENT SCORE : ");

System.out.println(p1.getName()+" 's Point score is "+ p1.getPoint()) ;

System.out.println(p2.getName()+" ' s Point score is "+ p2.getPoint()) ;

p1.Compare(p2);

}

}

if(secondDeck.size() == 0){ // Only when one of the player's deck is empty , winner will be declared

System.out.println();

JFrame f1 = new JFrame();

JOptionPane.showMessageDialog(f1, "game over\n"+" "+p1.getName()+" "+" wins the game");

break;

}

else if(firstDeck.size() ==0){

JFrame f2 = new JFrame();

JOptionPane.showMessageDialog(f2, "game over\n"+" "+p2.getName()+" "+" wins the game");

break;

}

//game over either one player runs out of card(deck size is 0)

}

}

//endwhile

public static void war(){ //war is a method that prints out the War message System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*WAR\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

}

}//end WarCardGame class