Project

<Casino War Card Game>

CIS-18A

Name: Aleksandar Videv

TABLE OF CONTENTS

**1. Summary…………………………………….………………………3**

1.1. How the Card Game Works**……………………………….…………4**

**3.The logic behind the game**……**…………………………….……...5**

3.1. Pseudo Code……**…………………………..……………………….5**

4. Program……**………………………………**……**………………..…..8**

5. Proof of a working product……**……………………………….….17**

**1. Summary**

The Casino War program is a game of chance in which the player places a wager against the dealer. The game is played using a standard deck of playing cards. The player starts with $1000 and can choose to place a wager on each round. If the player's card value is higher than the dealer's, the player wins and their money increases. If the dealer's card value is higher, the player loses and their money decreases. If the cards are a tie, the player has the option to surrender or go to war by doubling their wager. The game ends when the player has no more money or decided to leave.

The target audience for this program would be people interested in playing casino games, especially the game of War. The program has a simple and straightforward design, making it easy for players to understand the rules and start playing.

One of the strengths of this program is its ability to keep track of the player's money, wins, and losses, providing the player with an overall score for their performance. The use of the Scanner object for input also makes it easy for players to interact with the program.

However, there is room for improvement in terms of expanding the game's features. For example, the program could be enhanced by adding different types of bets, such as side bets or bonus rounds. Additionally, the program could be made more visually appealing by incorporating Integration of UI using Swing.

Overall, the Casino War program provides a basic and functional platform for playing the game of War, but there is potential for future expansion and improvement to make the game more enjoyable and engaging.

* 1. How the Card Game Works

Casino War is a card game that is typically played with a standard deck of 52 playing cards. The objective of the game is to have a higher card value than the dealer.

Each player, including the dealer, is dealt one card. The card with the highest value wins the round. If both cards have the same value, the game goes to "war." In war, the player has to double their wager, and then two more cards are dealt. The player with the higher card value among the two additional cards wins the war, and their wager is paid out accordingly.

If the player does not want to go to war, they have the option to "surrender," in which case they lose half of their wager.

**3.The logic behind the game**

3.1. Pseudo Code

Create deck of cards

Set player starting money = 1000

Set player wager = 0

Set number of wins = 0

Set number of losses = 0

WHILE player wants to keep playing

Ask player for their wager

IF wager is greater than player's money

Display "You don't have enough money to place that wager."

Continue to next iteration

IF wager is less than or equal to 0

Display "You can't wager less than $1."

Continue to next iteration

Deal cards for player and dealer

Display cards for player and dealer

IF player card value is greater than dealer card value

Increase player's money by wager

Add 1 to wins count

Display "You win! You now have $[player's money]."

ELSE IF player card value is less than dealer card value

Decrease player's money by wager

Add 1 to losses count

Display "You lose. You now have $[player's money]."

ELSE

Display "Tie!"

Ask player if they want to surrender or go to war

IF player chooses to surrender

Decrease player's money by half of wager

Add 1 to losses count

Display "You surrendered. You now have $[player's money]."

ELSE IF player chooses to go to war

Double wager

Deal additional cards for player and dealer

IF player's war card value is greater than dealer's war card value

Increase player's money by wager

Add 1 to wins count

Display "You win the war! You now have $[player's money]."

ELSE

Decrease player's money by wager

Add 1 to losses count

Display "You lose the war. You now have $[player's money]."

IF player's money is less than or equal to 0

Display "You have no money left. Game over."

Set playing to false

ELSE

Display "You won [number of wins] times and lost [number of losses] times."

Ask player if they want to keep playing

IF player answers no

Set playing to false

4. Program

import java.util.ArrayList;

import java.util.Collections;

import java.util.Scanner;

public class CasinoWar {

//deck of cards

private ArrayList<String> deck = new ArrayList<>();

//player's starting money

private int playerM = 1000;

//player's current wager

private int wager = 0;

//number of wins

private int wins = 0;

//number of losses

private int losses = 0;

//number of money won

private int moneyW = 0;

public static void main(String[] args) {

//create an instance of the CasinoWar game

CasinoWar game = new CasinoWar();

//start playing the game

game.play();

}

//constructor method that creates the deck of cards

public CasinoWar() {

createDeck();

}

//main logic for playing the game

public void play() {

//Scanner object for input

Scanner sc = new Scanner(System.in);

//flag to keep track of if the player wants to keep playing

boolean playing = true;

//loop to keep playing the game as long as the player wants to

while (playing) {

//ask the player for their wager

System.out.println("You have $" + playerM + ". Enter your wager: ");

wager = sc.nextInt();

//check if the player has enough money to place the wager

if (wager > playerM) {

System.out.println("You don't have enough money to place that wager.");

//continue to the next iteration of the loop

continue;

}

if (wager <= 0) {

System.out.println("You can't wager less than $1.");

//continue to the next iteration of the loop

continue;

}

//deal cards for the player and dealer

String playerCard = dealCard();

String dealerCard = dealCard();

//display the cards for the player and dealer

System.out.println("Your card is " + playerCard + ". The dealer's card is " + dealerCard + ".");

//check who wins or if it's a tie

if (getCardValue(playerCard) > getCardValue(dealerCard)) {

//player wins, increase their money and add to the wins count

playerM += wager;

wins++;

System.out.println("You win! You now have $" + playerM + ".");

} else if (getCardValue(playerCard) < getCardValue(dealerCard)) {

//player loses, decrease their money and add to the losses count

playerM -= wager;

losses++;

System.out.println("You lose. You now have $" + playerM + ".");

} else {

System.out.println("Tie!");

//ask the player if they want to surrender or go to war

System.out.println("Do you want to surrender (s) or go to war (w)?");

String response = sc.next();

if (response.equalsIgnoreCase("s") || response.equalsIgnoreCase("surrender")) {

//player surrenders, decrease their money by half and add to the losses count

playerM -= wager / 2;

losses++;

System.out.println("You surrendered. You now have $" + playerM + ".");

// Check if the player chose to go to war

} else if (response.equalsIgnoreCase("w") || response.equalsIgnoreCase("war")) {

// Double the wager

wager \*= 2;

System.out.println("You chose to go to war and doubled your stake to $" + wager + ".");

String playerWarCard = dealCard();

String dealerWarCard = dealCard();

System.out.println("Your war card is " + playerWarCard + ". The dealer's war card is " + dealerWarCard + ".");

// Determine the winner of the war

if (getCardValue(playerWarCard) > getCardValue(dealerWarCard)) {

playerM += wager;

wins++;

System.out.println("You win the war! You now have $" + playerM + ".");

} else {

playerM -= wager;

losses++;

System.out.println("You lose the war. You now have $" + playerM + ".");

}

}

}

// Check if the player has no money left

if (playerM <= 0) {

System.out.println("You have no money left. Game over.");

playing = false;

} else {

System.out.println("You won " + wins + " times and lost " + losses + " times.");

System.out.println("Do you want to keep playing? (y/n)");

String response = sc.next();

// Check if the player wants to keep playing

if (!response.equalsIgnoreCase("y") && !response.equalsIgnoreCase("yes")) {

playing = false;

}

}

}

moneyW = playerM - 1000;

if(playerM > 1000) {

System.out.println("You won " + wins + " times and lost " + losses + " times.");

System.out.println("You ended the game with $" + playerM + " and you have a gain $" + moneyW +".");

} else if(playerM < 1000) {

System.out.println("You won " + wins + " times and lost " + losses + " times.");

System.out.println("You ended the game with $" + playerM + " and you have lost $" + moneyW +".");

}

else{

System.out.println("You won " + wins + " times and lost " + losses + " times.");

System.out.println("You ended the game with $" + playerM + " and you have not lost or gain any money" + ".");

}

}

private void createDeck() {

// Declaring the suits of a deck of cards

String[] suits = {"Hearts", "Diamonds", "Spades", "Clubs"};

// Declaring the ranks of a deck of cards

String[] ranks = {"2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A"};

// Iterating over the suits array and ranks array to create a deck of cards

for (String suit : suits) {

for (String rank : ranks) {

// Adding each card to the deck list

deck.add(rank + " of " + suit);

}

}

}

private String dealCard() {

// Shuffling the deck to randomize the order

Collections.shuffle(deck);

// Removing and returning the first card from the deck

return deck.remove(0);

}

private int getCardValue(String card) {

// Extracting the rank of the card from the input string

String rank = card.split(" ")[0];

// Determining the value of the card based on its rank

switch (rank) {

case "A":

return 14;

case "K":

return 13;

case "Q":

return 12;

case "J":

return 11;

default:

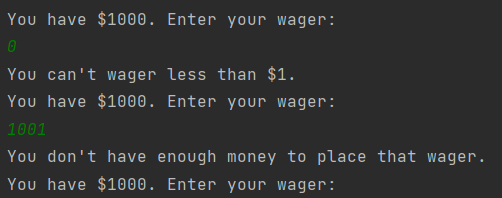
return Integer.parseInt(rank);

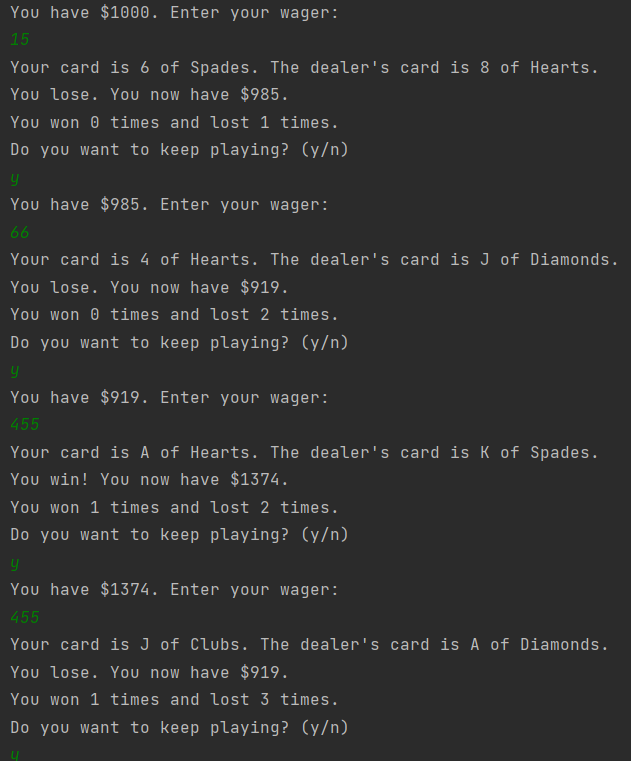
}

}

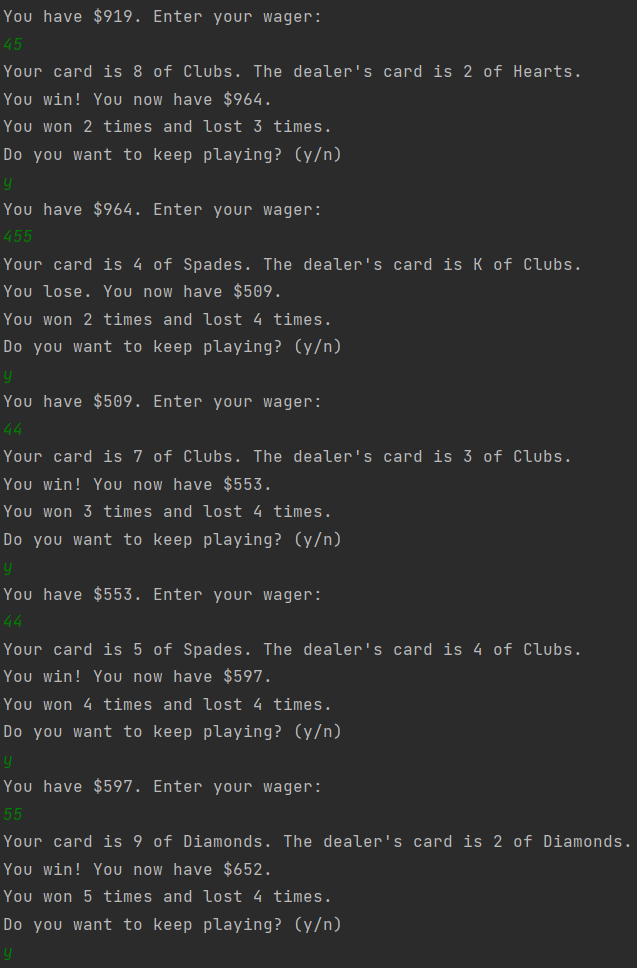
}

5. Proof of a working product

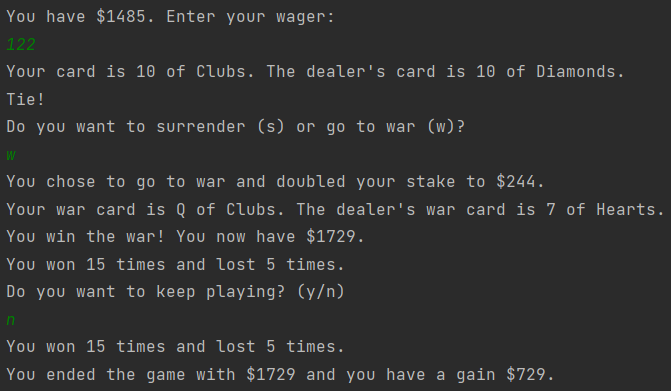
****

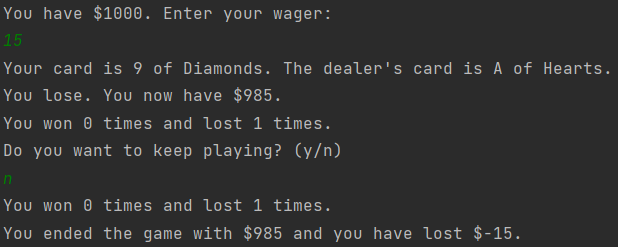
****

**…**

****

**…**

****

****