Blackjack Program

Yijun Chen

February 26, 2016

Abstract

Blackjack is a very common card game in gambling casinos, whoever gets the value that is closest to 21 wins the game. It is also known as "21". Now you can imagine that you are sitting in a casino in Las Vegas, the dealer is now distributing the first card to you. Wonder how far your luck could go? Place your wager and let's begin.

1 Representing Java Code in Your Assignment

The code can be divided into two parts. The first part contains all the method that imitated some actions in the process of dealing cards. It also set rules for the behavior of the cards.
The second part is the application of the methods above. And we can be

The count represented the number of cards remaining in the deck. This method shuffle the deck of cards. The integer 'switches' represents the number of

times we shuffle the cards. Now the cards has not been created yet, but we write the method first. This method just shuffle the strings(which representing all the 52 cards) again and again.
We first duplicate the "a-1"th element in the random deck, and place the replica to a temporary space we call 'temp'.And then we make the "b-1"th element in the deck equals to the "a-1"th element; finally, we make the "b-1"th element equals to "temp"(switch the position of these two). We can set the times of shuffling afterwards in the command line.

```
import java.util.*;
import java.util.Scanner;

public class Cards{

  static int count=52;

  public static int rand(int high){
      return (int) (high*Math.random()+1);}

  public static void shuffle(String[] the_deck, int switches){
    String temp;
    int a; int b;
    for(int i=0; i<switches; i++){
      a = rand(52);
      b = rand(52);
      temp = the_deck[a-1];
      the_deck[a-1] = the_deck[b-1];
      the_deck[b-1] = temp;}
  }
  public static String deal(String[] the_deck){
    count=count-1;
    return the_deck[count];}

  public static int aces(String the_card){
   if(the_card.charAt(0)=='A'){
       return 1;}
     else{
       return 0;}
   }
```

In blackjack, an ace has two different values. One is 1, and the other is 11.The player get the chance to choose the value of ace.We want to find the total number of aces in the "cards" Arraylist. Because we've created an array with 52 elements.

Now we need to identify all the 'Aces' in the array. Here what we do is to pull out the elements that begin with an "A".

In this method we set the value for each of the cards. We make the elements that begin with a "J", "Q", "K" or "1" to return number "10". If the element begins with the letter "A", the method will return "11"(but we alter it to "1" using the following method). Otherwise, the method returns the first thing in the element.

```
  public static int aces(ArrayList the_hand){
   int sum=0;
   for(int i=0; i<the_hand.size();i++){
     sum = sum + aces(the_hand.get(i).toString());}
   return sum;}

 public static int value(String the_card){
   char first = the_card.charAt(0);
   if (first=='1'|first=='J'|first=='Q'|first=='K'){
     return 10;
   }
     else if(first=='A'){
       return 11;}
     else{
     return Character.getNumericValue(first);}
   }
```

In this method, we identify the 'aces' in the Arraylist 'the hand' and sum up the elements that are chosen(cards that are distributed) in the Arraylist. If the sum(that includes an ace in it) has exceeded 21, then the method will turn the value of ace from 11 to 1.

```
 public static int value(ArrayList the_hand){
   int sum=0;
   int num_aces=aces(the_hand);
   for(int i=0; i<the_hand.size();i++){
     sum = sum + value(the_hand.get(i).toString());}
   while(num_aces>0 && sum>21){
     sum=sum-10;
     num_aces=num_aces-1;}
   return sum;
 }
```

This method is the last step to construct the Arraylist. The "suit" array represents the four categories of the cards: Clubs, Diamonds, Hearts and Spades.We use a nested for-loop to list all the elements; i is the four categories, j is the 13 elements under the four main branches. the first element in j is ace; the tenth element in j is Jack; the eleventh element is Queen; the twelfth element is King; the rest we just return the index number. And then we concatenate the category name with the card index. Finally, we shuffle the cards thoroughly. Two thousand time might be a good idea.

```
public static void main(String[] args){

  Scanner scan = new Scanner(System.in);

  String[] deck = new String[52];
  String[] suit = new String[4];
  int[] card = new int[13];

  for (int i=0; i<card.length; i++){
    card[i]=i+1;}
  String cardName;
  suit[0] = "Clubs";
  suit[1] = "Diamonds";
  suit[2] = "Hearts" ;
  suit[3] = "Spades";

  for(int i=0; i<4; i++){
    for(int j=0; j<13; j++){
      if(j==0){cardName="Ace";}
      else if(j==10){cardName="Jack";}
      else if(j==11){cardName="Queen";}
      else if(j==12){cardName="King";}
      else {cardName=Integer.toString(card[j]);}
      deck[ 13*i+j ]= cardName + "_" +suit[i];}
  }
    shuffle(deck, 2000);
```

Here is part the game really begins. We create two Arraylists(the utility of an arraylist is that we can easily add to the array like we do in Python) for both the dealer and the player as the pockets they keep their cards. After the dealer have given himself two cards, the player now get one card from the dealer. The dealer will ask if the player want "Hit" or "Stay".

While the player answers "Hit"(which we use the abbreviation "H" in the code), the dealer will give one more card to the player. The dealer only stops distributing card to the player under two circumstances: when the player says "Stay" or when the sum of cards in the player's pocket(which is the Arraylist "hand") has exceeded 21. The player can choose whether to stay or hit as long as the sum is less than 21. The dealer, however, has to hit if the sum is less than 17. ALL the choose operations will be done in the interaction pane(or through the command line). In the end, if the sum of player is greater than the dealer's sum(and the player's sum is less than 21) or the dealer's sum is greater than 22, the method will print out "YOU WIN!!!". If the player's sum is greater than the dealer's sum or the player's sum is less than the dealer's sum(and the dealer's sum is less than 21), the method will print out the string "YOU LOSE. BOO!!!"

```
  String say;
boolean state=true;

  ArrayList hand = new ArrayList();
  ArrayList dealer_hand = new ArrayList();
  dealer_hand.add( deal(deck) );
  dealer_hand.add( deal(deck) );
  hand.add( deal(deck) );

  while(state){

  hand.add( deal(deck) );

  System.out.println("Dealer showing: " + dealer_hand.get(1));
  System.out.println("Contents of hand: " + hand);
  System.out.println("Your score is: " + value(hand));

  if(value(hand)>21){
    System.out.println("BUST!!!!");
    break;
  }

  System.out.println( "hit[H] or stand[S]?");
       say=scan.nextLine();
       if(say.equals("H")){state=true;}
       else{state=false;}
  }

  while( value(dealer_hand)<17 ){
    dealer_hand.add( deal(deck) );
  }
```

```
System.out.println("Dealer has: " + dealer_hand);
System.out.println("Dealer score is: " + value(dealer_hand));

if( (value(hand)>value(dealer_hand) && value(hand)<22) | (value(dealer_hand) > 21) ){
  System.out.println( "YOU WIN !!!!");
}
else{System.out.println( "YOU LOSE. BOO !!!!");}
```