Homework #4 POOCasino Blackjack

B01902080 資工四 王于青

1. Implementations of my player's strategy

(1) Making bet:

For each round, my initial bookie is one fifteenth of the current chips I hold.

Then, I use a random multiplier to slightly change the bookie based on human actions.

If player wins in the last round, the multiplier falls between [1, 2].

If player pushes in the last round, the multiplier falls between [0.75, 1.5].

If player loses in the last round, the multiplier falls between [0.5, 1].

(2) Buying insurance:

I make this decision simply based on the probability of dealer getting a 10.

The probability for dealer to get a 10 roughly falls between [0.186, 0.372].

Therefore, player will buy insurance if the probability this round is higher than 0.28.

(3) Doing split, double, hit, and stand:

I use the general one-deck strategy below to implement player's decision.

I create several lookup tables to store the decisions under different circumstances.

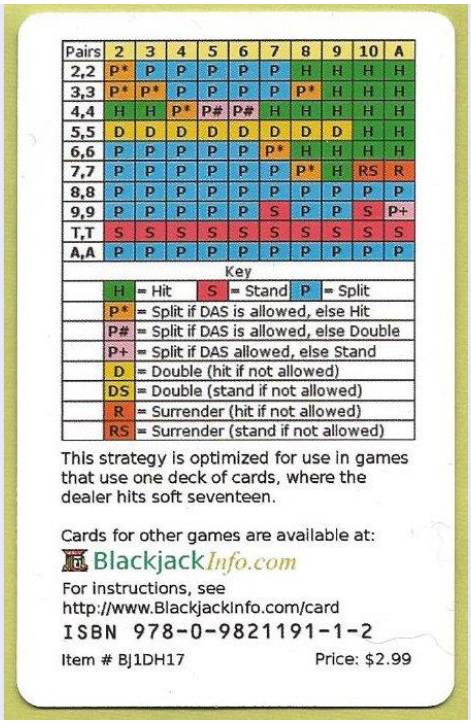
Then, I can simply calculate the index of [my_open, dealer_open] pairs and return the corresponding action in the lookup tables.

Blackjack Basic Strategy Chart
1 Deck, Dealer Hits Soft 17

	Dealer Upcard									
Hard Total	2	3	4	5	6	7	8	9	10	A
5-7	H	H	H	H	H	H	H	H	H	H
8	H	H	H	D	D	H	H	H	H	H
9	D	D	D	D	D	H	H	H	H	H
10	D	D	D	D	D	D	D	D	H	H
11	D	D	D	D	D	D	D	D	D	D
12	H	H	S	S	S	H	H	H	H	H
13	S	S	S	S	S	H	H	H	H	H
14	S	S	S	S	S	H	H	H	H	H
15	S	S	S	S	S	H	H	H	H	R
16	S	S	S	S	S	H	H	H	R	R
17	S	S	S	S	S	S	S	S	S	RS
Soft Total	2	3	4	5	6	7	8	9	10	A
A,2	H	H	D	D	D	H	H	H	H	H
A,3	H	H	D	D	D	H	H	H	H	H
A,4	H	H	D	D	D	H	H	H	H	H
A,5	H	H	D	D	D	H	H	H	H	H
A,6	D	D	D	D	D	H	H	H	H	H
A,7	S	DS	DS	DS	DS	S	S	H	H	H
A,8	S	S	S	S	DS	S	S	S	S	S
A,9	S	S	S	S	S	S	S	S	S	S

(Pairs are listed on back of card.)

Ken Smith's BlackjackInfo.com



Pairs	2	3	4	5	6	7	8	9	10	A
2,2	P*	P	P	P	P	P	H	H	H	H
3,3	P*	P*	P	P	P	P	P*	H	H	H
4,4	H	H	P*	P#	P#	H	H	H	H	H
5,5	D	D	D	D	D	D	D	D	H	H
6,6	P	P	P	P	P	P*	H	H	H	H
7,7	P	P	P	P	P	P	P*	H	RS	R
8,8	P	P	P	P	P	P	P	P	P	P
9,9	P	P	P	P	P	S	P	P	S	P+
T,T	S	S	S	S	S	S	S	S	S	S
A,A	P	P	P	P	P	P	P	P	P	P

Key

H	= Hit	S	= Stand	P	= Split

P*	= Split if DAS is allowed, else Hit
P#	= Split if DAS is allowed, else Double
P+	= Split if DAS allowed, else Stand
D	= Double (hit if not allowed)
DS	= Double (stand if not allowed)
R	= Surrender (hit if not allowed)
RS	= Surrender (stand if not allowed)

This strategy is optimized for use in games that use one deck of cards, where the dealer hits soft seventeen.

Cards for other games are available at:

BlackjackInfo.com

For instructions, see
http://www.BlackjackInfo.com/card

ISBN 978-0-9821191-1-2

Item # BJ1DH17 Price: $2.99

(4) Doing surrender:

As for surrendering, since the player cannot know the value of the face-down card, it's not possible for player to follow the actions in the lookup table.

Therefore, I assume the face-down card is a 10-value card (probability of 10 is usually the highest).

Then, player can follow the lookup table above to see if surrendering is necessary.

2. Designs and reasons of all the classes related to the casino

(1) Shuffler.java

I implement a Shuffler with methods to shuffle one deck of cards and assign single card.
Then I can simply shuffle cards and assign cards at the beginning of each round, and get new cards when players do splitting and hitting with this class.

(2) POOCasino.java

a. I make all dealer instructions specified in spec into functions (as the screenshot shown below), and then I do further functionizing and implementations inside each function.

```
System.out.printf("####### Round %d #######\n\n", curRound);

            clearVariables();
            shuffleOneDeck();

            askBets();

            if (indexMatcher.size() == 0) {
System.out.printf("All players are out of the game.\n");
                break;
            }

            assignInitialCards();
            assignCardsbyCommands(args);

            if (isDealerFaceUpACE())
                askInsurances();

            if (!isDealerBlackjackWithHoleCard()) {
                askSurrenders();

                for (int i = 0; i < indexMatcher.size(); i++) {
System.out.printf(">> Player%d's turn <<\n", indexMatcher.get(i)+1);
                    //skip the player who surrenders
                    if (isSurrender(indexMatcher.get(i)))
                        continue;

                    flipUpFaceDownCard(i);

                    Card card1 = hands.get(i).getCards().get(0);
                    Card card2 = hands.get(i).getCards().get(1);
                    if (isEqualFaceValue(card1, card2) && isFirstSplit(i)) {
                        if (askSplit(i, indexMatcher.get(i))) {
                            i--;
                            continue;
                        }
```

By doing so, it's easier for me to check if the existing bugs and unexpected outcomes are caused by logical mistakes or programming errors.

b. I use an ArrayList<Integer> indexMatcher to specify the relation between hands and players. Therefore, I can still get the owner of each hand

1. when some players are broke and out of the game.
2. after some players do splitting and have two hands at the same time.

c. Since the given Player.class requires specific type of inputs (e.g., Card, ArrayList<Card>, Hand…), I implement some methods to convert objects between these types efficiently.

(3) PlayerB01902XXX.java toString()

The toString() method of my classmates' and mine will print the remaining chips the player holds. Then, it would be easier for us to keep track of players' accounts during the game and therefore check the correctness of our program.

3. The results of the duel between my classmates and me

Player1: PlayerB01902080 (my player) , Player2: PlayerB01902041

Player3: PlayerB01902102 , Player4: PlayerB01902058

(1) Result of the whole game (partial screenshot is shown as below):

The result of 1st round

```
POOCasino Blackjack, written by b01902080 Yu-Ching Wang.

There will be 1 round(s) Blackjack game.
Each player starts with initial chips 1000.
####### Round 1 #######

>> Start betting <<
Player1 bets 50 chip(s).
Player2 bets 10 chip(s).
Player3 bets 10 chip(s).
Player4 bets 100 chip(s).

>> Assign cards <<
Player1 gets a face-up card HK.
Player2 gets a face-up card D3.
Player3 gets a face-up card S4.
Player4 gets a face-up card C10.
Dealer gets a face-up card H9.

Dealer does not get a Blackjack.
>> ask surrender <<

>> Player1's turn <<
Player1 gets a hand HK DA.
[Stand]

>> Player2's turn <<
Player2 gets a hand D3 S7.
[Hit] Player2's hand becomes D3 S7 S5.
[Hit] Player2's hand becomes D3 S7 S5 CK.
[Stand]

>> Player3's turn <<
Player3 gets a hand S4 DQ.
[Hit] Player3's hand becomes S4 DQ D7.
[Stand]

>> Player4's turn <<
Player4 gets a hand C10 S10.
Player4 gets a pair.
Player4 does not choose to split.
[Stand]

>> Dealer's turn <<
Dealer gets a hand H9 H8.
[Stand]

>> Compare hands <<
Player1: 21 (Blackjack)
Player2: 25
Player3: 21
Player4: 20
Dealer: 17

[Player1] Player1 gets a Blackjack. 75 chip(s) goes to the player.
[Player2] Player2 gets busted. 10 chip(s) goes to the casino.
[Player3] Player3 gets more face values. 10 chip(s) goes to the player.
[Player4] Player4 gets more face values. 100 chip(s) goes to the player.

>> Check insurance <<

Player1 has 1075.0 chip(s) in hand.
Player2 has 990.0 in hand.
Player3 has 1010.0 chips in hand.
Player4 has 1100.0 chips in hand.
```

The result of last (1000st) round

```
####### Round 1000 #######

>> Start betting <<
Player1 bets 4451 chip(s).
Player2 bets 10 chip(s).
Oops! Player3 is broke by -1125.0.
Player3 has no money to bet 1280 chip(s).
Player4 bets 29 chip(s).

>> Assign cards <<
Player1 gets a face-up card D7.
Player2 gets a face-up card H3.
Player4 gets a face-up card S5.
Dealer gets a face-up card S8.

Dealer does not get a Blackjack.
>> ask surrender <<

>> Player1's turn <<
Player1 gets a hand D7 S10.
[Stand]

>> Player2's turn <<
Player2 gets a hand H3 D9.
[Hit] Player2's hand becomes H3 D9 H8.
[Stand]

>> Player4's turn <<
Player4 gets a hand S5 D5.
Player4 gets a pair.
Player4 does not choose to split.
[Double Down]
Player4's bet becomes 58 chip(s).
Player4's hand becomes S5 D5 H7.
[Stand]

>> Dealer's turn <<
Dealer gets a hand S8 CA.
[Stand]

>> Compare hands <<
Player1: 17
Player2: 20
Player4: 17
Dealer: 19

[Player1] Dealer gets more face values. 4451 chip(s) goes to the casino.
[Player2] Player2 gets more face values. 10 chip(s) goes to the player.
[Player4] Dealer gets more face values. 58 chip(s) goes to the casino.

>> Check insurance <<

Player1 has 33275.0 chip(s) in hand.
Player2 has 1280.0 in hand.
Player3 has 155.0 chips in hand.
Player4 has 241.0 chips in hand.

Blackjack game over. The result is as below:

Player1 has 33275.0 chip(s) in hand.
Player2 has 1280.0 in hand.
Player3 has 155.0 chips in hand.
Player4 has 241.0 chips in hand.

Thank you for your playing :)
See you next time.
```

(2) Make bet (handling player.brokeException):

```
>> Start betting <<
Player1 bets 54 chip(s).
Player2 bets 10 chip(s).
Oops! Player3 is broke by -1080.0.
Player3 has no money to bet 1280 chip(s).
Player4 bets 6 chip(s).
```

(3) Buy insurance:

```
Dealer gets an ACE face-up card <3
Player1 buys an insurance of 50 chip(s).
Player2 buys an insurance of 5 chip(s).
Player4 does not buy an insurance
```

(4) Do surrender:

```
Dealer does not get a Blackjack.
>> ask surrender <<
Player1 chooses to surrender.
```

(5) Do split:

```
>> Player2's turn <<
Player2 gets a hand S10 C10.
Player2 gets a pair.
Player2 chooses to split.

>> Player2's turn <<
Player2 gets a hand S10 H8.
[Stand]

>> Player2's turn <<
Player2 gets a hand C10 H4.
[Hit] Player2's hand becomes C10 H4 DA.
[Hit] Player2's hand becomes C10 H4 DA C5.
[Stand]
```

(6) Do double down:

```
>> Player2's turn <<
Player2 gets a hand H2 S9.
[Double Down]
Player2's bet becomes 20 chip(s).
Player2's hand becomes H2 S9 D5.
[Stand]
```

(7) Do hit & stand:

```
>> Player2's turn <<
Player2 gets a hand DA H5.
[Hit] Player2's hand becomes DA H5 D7.
[Hit] Player2's hand becomes DA H5 D7 C7.
[Stand]
```

(8) Dealer's actions

```
>> Dealer's turn <<
Dealer gets a hand SA S6.
[Hit] Dealer's hand becomes SA S6 CJ.
[Stand]
```

```
>> Dealer's turn <<
Dealer gets a hand HA S5.
[Hit] Dealer's hand becomes HA S5 S10.
[Stand]
```

(9) Compare the dealer's and players' hands

```
>> Compare hands <<
Player1: 15
Player2: 20
Player3: 21 (Blackjack)
Player4: 6
Dealer: 21 (Blackjack)

[Player1] Dealer gets a Blackjack. 46 chip(s) goes to the casino.
[Player2] Dealer gets a Blackjack. 10 chip(s) goes to the casino.
[Player3] Player3 gets a Blackjack. Push.
[Player4] Dealer gets a Blackjack. 81 chip(s) goes to the casino.

>> Check insurance <<
Player1 buys an insurance. 46 chip(s) goes to the player.
Player2 buys an insurance. 10 chip(s) goes to the player.
Player4 buys an insurance. 81 chip(s) goes to the player.
```

(10) What we've learned during the duel:

My classmates say that I'm a cautious player compared to other players.

My bet is never bigger than 2/15 of the current chips I hold, so I won't be broke anyway.

On the other hand, some of my classmates will make bigger bets intending to win back their loss regardless of how many chips they have. (isn't that too crazy 😐 ?)

Therefore, my classmates sometimes become billionaires while sometimes they get broke.

I usually earn less money than they do, but I won't be broke and I can leave the casino safely <3

4. BONUS TIME <3

(1) Exceptions checking:

Whenever player is doing actions related to money (i.e., increase_chips(diff), decrease_chips(diff)), I will check if the action is valid and do the following instructions:

a. brokeException when making bet: player is out of the current round.
   (Side effect: player who is broke will be out of the rest of the game.)
b. brokeException when buying insurance: player is not allowed to buy insurance.
c. brokeException when doing split: player is not allowed to split hand.

Also, if player make a 0 bet, I assume the player is quitting the game.

(2) Assigning target cards commands:

By modifying makefile make run commands, user can assign target cards to players and dealer.

Commands: -$(playerPosition)$(openCardSuit)$(openCardValue)$(holeCardSuit)$(holeCardValue)

Ex: java POOCasino … playerB01902XXX -1SAHT -2D2D5 -3CQS8 -4H7D7 –dCADK

Player1 will get a hand SA H10.

Player2 will get a hand D2 D5.

Player3 will get a hand CQ S8.

Player4 will get a hand H7 D7.

Dealer will get a hand CA DK.

With these commands, it's more convenient and efficient to test the correctness of my program.

(3) BLACKLIST mechanism:

I've heard from other casinos that there exists a difficult customer B01902135 who usually makes negative bets, does split and double when having not enough money, and throws exceptions whenever she likes without considering the dealer and other players.

In order for the fairness and safety of my lovely casino, I've listed the customer B01902135 on my blacklist and forbidden the customer to come into my casino FOREVER <3