COP 3502 Project 1 – Black Jack

Overview

The blackjack project is designed to improve student's elementary abilities with data types, print statements, Java libraries and decision-making structures (if-statements with an introduction to loops). It is also meant to be playful and practical; to engage students to real-life coding.

Rules of the Road

Just a forewarning, this program that we are creating is NOT how true blackjack is played at your typical casino. Don't drop out of your studies and move to Vegas just because you're owning at this game.

A typical game will play out as follows:

The player will be dealt one card at the start of the game (a game means one round of play). The player, based off the value of his card collection, can either ask for another card (a hit) or choose to hold (stand). The dealer will then begin his turn and try to beat your hand. The player is competing against the dealer, who also has his own hand. Whomever is closer to 21 at the end of the game (as long as they don't exceed 21) wins the game.

1. Player's turn: try to get as close to or achieve 21 without going over. (Getting to 21 is the best a player can do)
2. Dealer's turn: Tries to beat the Player's hand without going over 21.
3. Decide who wins after the game is done.
4. Increment the score for either the dealer or the player based on whoever won.
5. Repeat!

You will need to use a while loop in your program. A loop will allow you to keep playing successive games without restarting the program each time and will also allow you to keep score for a number of games.

Here is the basic syntax of a while loop:

```
while (Boolean expression)
{
   // Statements
}
```

Inside the while parentheses will be a boolean expression. If it evaluates to true, then we continue executing the loop. If that statement ever evaluates to false, then we stop executing the loop (skip over the loop block) and keep continuing down the line of our code.

Here is an example of a while loop:

```
int x= 0;
while (x < 17)
{
   x += 10;
}
```

The variable x is initialized to zero and then the conditional statement is checked. Since 0 is less than 17, the conditional statement evaluates to true and the statement in the while loop block will be executed. The number 10 is added to the value of x, so x is now equal to 10. We now have executed all the statements in the loop block so we jump back up to the conditional statement. The expression evaluates to true since 10 is less than 17, so we execute the statement in the loop again. Now x is equal to 20. Again we have executed all the statements in the loop and we jump back up to the conditional statement. Since 20 is not less than 17, the expression evaluates to false and the loop block will be skipped. Then any code after the while loop will be executed.

In this project, you will want your loop to continue executing until the player chooses the exit option from the menu.

Structure

Now that you have a general overview, here are the nitty gritty details.

When you start your program you should print “START GAME #1” to the console and the player will automatically be dealt their first card. With each new game played print the corresponding number.

To determine what card the player is dealt, you will need to generate a random number between 1 and 13 and add the card to the player’s hand.

The range of random generation will be from 1-13 which means….
The values correspond to these cards:

1	ACE!
2 - 10	correspond to their inherent values.
11	JACK!
12	QUEEN!
13	KING!

Face cards (King, Queen, Jack) are worth a value of 10. Therefore if the player is dealt a King (generated as 13), you will need to make sure the value 10 gets added to the player’s hand rather than a 13. Aces are going to be worth a value of 1. Every other card will be worth its number value.

Whenever a card is dealt, print the value of the card and the total value of your hand. (See sample output for format). If the card you were dealt was a face card or an ace then print the type of card. For example, if you were dealt a King you would print ***“Your card is a KING!”*** If the card was not a face card or an ace print the value of the card. For example, ***“Your card is a 2!”***.

After you were dealt a card, print the menu to the player. The menu should look like this:

1. Get another card
2. Hold hand
3. Print game statistics
4. Exit

If the player chooses option 1, the player will be dealt another card. If the player has a hand of 21, they automatically win and ***"BLACKJACK! You win!"*** is printed**.** Then a new game is started. If the player's hand exceeds 21, the dealer automatically wins and a new game is started. In this case print, ***"You exceeded 21! You lose :(".***

If the player chooses to hold their hand (option 2), then the dealer will be dealt his hand. To determine the dealer's hand, generate a random number between 16 and 26 (both inclusive).

If the dealer's hand is above 21, the player automatically wins. If both the dealer and player have the same value hand then no one wins. In this case print ***"It's a tie! No one wins!"***. Otherwise, whoever has the higher hand value wins the round. If the player wins, print ***"You win!"***. If the dealer wins print ***"Dealer wins!"***
After the winner (or tie) has been determined a new game is started.

If the player chooses option 3, you will print the statistics of the game. Throughout your program you will need to keep track of the number of games played, the player's number of wins, the dealer's' number of wins and the number of ties. Print out all these values as well as the percentage of player wins to the total number games played. Format your percentage value to one decimal point. See sample output for format.

If option 4 is chosen then exit the program.

If any other input is entered, print:

"Invalid input!
Please enter an integer value between 1 and 4."

Then redisplay the menu. Your program should be able to handle any input that the user enters without throwing an exception.

Note: "When a scanner throws an InputMismatchException, the scanner will not pass the token that caused the exception, so that it may be retrieved or skipped via some other method." This means that when the scanner throws the InputMismatchException on an incompatible data type the scanner will not clear the incompatible input and you will not be able to enter input again unless you clear the scanner. Therefore, after the error is caught, you will need to clear the scanner by using scanner.nextLine().

Submissions

*Output needs to be formatted exactly like the provided sample output.

Please name your file BlackJack.java.

Submit your zipped src folder at the end of this project. Do not submit any other files.

Sample Output

*Please note that you will be generating card values randomly so your numbers will be different

```
START GAME #1

Your card is a 4!
Your hand is: 4

1. Get another card
2. Hold hand
3. Print statistics
4. Exit

Choose an option: 1

Your card is a 9!
Your hand is: 13

1. Get another card
2. Hold hand
3. Print statistics
4. Exit

Choose an option: 1

Your card is a 7!
Your hand is: 20

1. Get another card
2. Hold hand
3. Print statistics
4. Exit

Choose an option: 2

Dealer's hand: 22
Your hand is: 20

You win!

START GAME #2

Your card is a KING!
Your hand is: 10

1. Get another card
2. Hold hand
3. Print statistics
```

```
4. Exit

Choose an option: 1

Your card is a 7!
Your hand is: 17

1. Get another card
2. Hold hand
3. Print statistics
4. Exit

Choose an option: 2

Dealer's hand: 17
Your hand is: 17

It's a tie! No one wins!

START GAME #3

Your card is a 7!
Your hand is: 7

1. Get another card
2. Hold hand
3. Print statistics
4. Exit

Choose an option: 1

Your card is a 10!
Your hand is: 17

1. Get another card
2. Hold hand
3. Print statistics
4. Exit

Choose an option: 2

Dealer's hand: 20
Your hand is: 17

Dealer wins!

START GAME #4

Your card is a 4!
Your hand is: 4
```

```
1. Get another card
2. Hold hand
3. Print statistics
4. Exit

Choose an option: 6

Invalid input!
Please enter an integer value between 1 and 4.

1. Get another card
2. Hold hand
3. Print statistics
4. Exit

Choose an option: 1

Your card is an ACE!
Your hand is: 5

1. Get another card
2. Hold hand
3. Print statistics
4. Exit

Choose an option: 1

Your card is a JACK!
Your hand is: 15

1. Get another card
2. Hold hand
3. Print statistics
4. Exit

Choose an option: 1

Your card is a 10!
Your hand is: 25

You exceeded 21! You lose :(

START GAME #5

Your card is a 6!
Your hand is: 6

1. Get another card
2. Hold hand
```

```
3. Print statistics
4. Exit

Choose an option: 3

Number of Player wins: 1
Number of Dealer wins: 2
Number of tie games: 1
Total # of games played is: 4
Percentage of Player wins: 25.0 %

1. Get another card
2. Hold hand
3. Print statistics
4. Exit

Choose an option: 3.5

Invalid input!
Please enter an integer value between 1 and 4.

1. Get another card
2. Hold hand
3. Print statistics
4. Exit

Choose an option: 4
```