The BlackJack vignette

```
#Load the library
library(BlackJack)
#readytoplay()
```

Introduction

Have you ever had a long night of school work and need something fun to blow off some steam? What about when you were home on a Friday night alone, because all of your buddies are out with their significant others? Or if you are a card-shark in the making, and just want to get some practice in.

Well this is the package just for you. Black jack gives you some of the best odds out of all of the casino games. This package is for when you aren't at that environment, and maybe want a little practice. It never hurt anybody, and in the process you can take some of your friends' money the next time they are free on a Friday. The package is really simple in design. We wanted to bring together an easy game of black jack. We want to offer the user an outlet to practice their skills if they choose. There are just two players. Yourself and the dealer. Because that's what really counts. So sit back, crack a cold one, relax, and let's play some good old black jack.

House Rules

Before diving into the game itself let's cover the ground rules. The ultimate objective of this game is to have a higher score than the dealer's score (if you get 21, that's just a nice bonus). It is critical you remember you are in competition with the dealer. They will also be aiming to achieve a score of 21. Blackjack begins with the player receiving the deal of two cards. The dealer will deal themself two cards as well, one will be face up (up card) and one will be facing down (hole card). Now you might be asking, how do the cards lead to a score of 21. This is a relatively straight forward and simple concept with one little twist. Face cards (Jack, Queen, King) have a value of 10 points. For example, say we are dealt an initial hand of a King and a 5. Our total points for this initial hand would be 15. The one twist is the Ace. Aces can either hold a value of 11 or 1. Depending on the scenario to "benefit" the player. For example, if player is dealt a Jack and an Ace, that's a blackjack (21)! If a player is dealt an ace and a 2, that player can choose to have their score be a 13 or 3. As the player, you are allowed to take as many hits (take another card) as you would like. One trick of the game lies in not "busting" or going over the allowed max score of 21. You want to be confident that your score will beat the dealers and remember the dealer has some rules they have to follow. In the BlackJack package our dealer will be staying at 17. Meaning if they have a score that is greater than or equal to 17 that is the dealer's final score. If the dealer has anything below the score of 17, they will be required to take a hit. Remember these rules and use them to your advantage!

The BlackJack package took a page out of the casinos' book with a slight twist. Here we added the freedom for you to decide on how many decks you'd like to play with. The popular casino choice is 6 decks, for a total of 312 cards. If that's how you want to play it, just input that into the function below. It makes little difference to us how you want to play it, we will have it set up for you either way.

```
#Begin by inputting the number of Decks you would like to play with.
Deck1<- Build_playing_deck(5)
```

```
#Just a quick check to make sure everything is looking correct.
head(Deck1)
```

```
##    Suits Cards
## 1 Hearts   Ace
## 2 Hearts     2
## 3 Hearts     3
## 4 Hearts     4
## 5 Hearts     5
## 6 Hearts     6
```

```
tail(Deck1)
```

```
##     Suits Cards
## 255 Clubs     8
## 256 Clubs     9
## 257 Clubs    10
## 258 Clubs  Jack
## 259 Clubs Queen
## 260 Clubs  King
```

We begin

Alright, we have the deck's inputted into the system. Now we can begin the game. The dealer will go ahead and deal you a hand all you have to do is ask for one. We will show you how to do that below.

```
#Input the deck being used into the MyHand function
my_hand<- MyHand(Deck1)
```

```
##     Suits Cards
## 41 Hearts     2
## 45 Hearts     6
```

Okay, that's a start. We now know what we are holding in our hand, let's find out what the dealer has going on. Here we will have the dealer show us the visible card in his hand (up card). We will keep his second card (hole card) hidden from view.

```
#Input the deck of the game into the Up_card function
Upcard1<- Up_card(Deck1)
```

```
##        Suits Cards
## 102 Diamonds  Jack
```

Alright, the dealer is sitting pretty solid. We know that our score is not looking too good compared to the dealer's hand here. So, we will want to go ahead and take another card (hit). So dealer, hit us!

```
#Here we input the deck used along with the hand we were dealt
my_cards1<- Hit(Deck1, my_hand)
```

```
##       Suits Cards
## 41   Hearts     2
## 45   Hearts     6
## 90 Diamonds Queen
```

Alright, we know what we have now. Feeling confident? Let’s see what the dealer has.

```
Dealers_cards<- Dealers_hand(Deck1, Upcard1)
```

```
##        Suits Cards
## 102 Diamonds  Jack
## 14    Hearts   Ace
```

Alright that’s the dealers hand. Let’s see if we made the right decision taking a hit. With this current hand the dealer is below the amount required for them to stand (17). Remember if the dealer has a score over 17, then they are required to stand. That will be their final score for the round.

```
Dealers_cards1<- Hit(Deck1, Dealers_cards)
```

```
##        Suits Cards
## 102 Diamonds  Jack
## 14    Hearts   Ace
## 159   Spades     3
```

Alright that’s the hand! Don’t worry, there’s more! We will play one more this time with less explaining and just playing. But first we should shuffle our deck.

```
#add new deck by utilizing shuffle function and build playing deck function
Deck2<- Shuffle(Build_playing_deck(4))
head(Deck2)
```

```
##      Suits Cards
## 206  Clubs  Jack
## 192  Clubs    10
## 17  Hearts     4
## 31  Hearts     5
## 111 Spades     7
## 123 Spades     6
```

Okay the deck has been shuffled. Let’s start up a new hand!

```
#Input Deck2 into MyHand function
my_hand1<- MyHand(Deck2)
```

```
##      Suits Cards
## 150 Spades     7
## 120 Spades     3
```

That’s interesting, let’s see the dealer’s up card.

```
#Input Deck2 into Up_card function
Upcard2<- Up_card(Deck2)
```

```
##     Suits Cards
## 25 Hearts Queen
```

Okay, let’s go ahead and take a hit for fun! (This may not be advised in a real game)

```
#Input Deck2 and my_hand1 into the hit function
my_cards2<- Hit(Deck2, my_hand1)
```

```
##         Suits Cards
## 150    Spades     7
## 120    Spades     3
## 87.1 Diamonds     9
```

Let’s hold here, and see what the dealer has going on.

```
#Input Deck2 and the dealer's up card (Upcard2)
Dealers_cards2<- Dealers_hand(Deck2, Upcard2)
```

```
##     Suits Cards
## 25 Hearts Queen
## 39 Hearts  King
```

Alright! One more hit on the dealer!

```
#Input Deck2 and the dealer's cards (Dealers_cards2)
Dealers_cards3<- Hit(Deck2, Dealers_cards2)
```

```
##          Suits Cards
## 25      Hearts Queen
## 39      Hearts  King
## 101.2 Diamonds    10
```

```
Dealers_cards4<- Hit(Deck2, Dealers_cards3)
```

```
##          Suits Cards
## 25      Hearts Queen
## 39      Hearts  King
## 101.2 Diamonds    10
## 152.1   Spades     9
```

Finally! We have gone through and played two hands of BlackJack. This concludes my demonstration of the BlackJack package. We have a package built for the everyday user of R. The idea behind this package was too keep it simple and just have a fun time. This vignette walks you through the basics of how to play the game while simultaneously operating the functions within the package.

There are quite a couple pitfalls associated with this package, when I do a self-reflection. The largest reason for the lack of having them done is there just wasn't enough time in the day for me to bring this to where I ideally would like it. As you may have noticed, this is lacking extra games. That would be my first priority to solidify this project. Beginning with adding the game of spades, followed by hearts, euchre, hold'em, etc. This package has a ton of potential. It could be a great time for anyone looking to just have an escape. The plan of action to implement these would be, write out the game rules and model the functions within the package after the separate games. Similar to how I have done it with the BlackJack package.

The second large scale improvement would be too implement a better shiny object. This was not a strong point of mine. In the future it would be cool to sit down and have the ability to see an actual playing card surface within a shiny application. After plenty of hours searching, I realized this was beyond my scope of knowledge at the moment. A way for me to potentially combat this in the future would be to dive head first into learning about shiny objects.

Overall, this project took a lot of my time (even if it doesn't look like it would). I have had a very bittersweet relationship with it, but I hope it can bring others some enjoyment. So please, this round's on the house.

Link to Github: Even though it's broken. https://github.com/connerm12/BlackJack