Black Jack

Aidan Fearn

Bellarmine University

CS-150 | Python

Abstract:

The goal behind my project is to provide the user with the ultimate blackjack gaming experience that you would receive from a casino with the ability to play from the comfort of their own home.

Introduction:

In a world where we are starting to see a major shift to a digital era, it is important that we have innovations that adapt to the new demands for simplicity and ease. That's why I have decided to create the ultimate black jack program, so that users can skip the trip to the casino and play their favourite casino games from the comfort of their own home.

Methodology:

Users will be able to use certain features like 'hit', 'stand', 'double-down' and 'split' in order to improve their odds and beat the house! The blackjack simulator will simulate a casino experience and have the same rules that you would find at a normal blackjack table. The objective of the game is to get as close as possible to 21 without going over or "busting" where the cards represent their respective face value (ex: a card "5 of spades" represents a numerical value of 5) with the exception of any Jacks, Queens and Kings where they will represent a numerical value of 10. Aces are also an exception where they will represent a numerical value of either 1 or 11. If the dealer beats the player, the message will display "you lose" and if the player beats the dealer, it will display a message saying "you win".

Some key python features that will be utilised will be the lists (in order to define cards to a given suit and numerical value) and the random function in order to display random cards every time the user runs the code. There will also be the implementation of while loops and if statements in order to dictate the running sequence that the user may or may not choose to engage in. Below is an example:

```
# Import the 'random' module to shuffle cards randomly

import random

# Define the suits, ranks, and values for the cards

suits = ["Hearts", "Diamonds", "Clubs", "Spades"]

ranks = ["Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine", "Ten", "Jack", "Queen", "King", "Ace"]

values = [2, 3, 4, 5, 6, 7, 8, 9, 10, 10, 10, 10, 11]
```

```
# Function to deal a random card

def deal_card():

    suit = random.choice(suits)

    rank = random.choice(ranks)

    value = values[ranks.index(rank)]

    return (rank, suit, value)

# Function to calculate the total value of a hand, considering aces

def calculate_hand_value(hand):

    value = sum(card[2] for card in hand)

    num_aces = sum(1 for card in hand if card[0] == "Ace")

while value > 21 and num_aces:

        value -= 10

        num_aces -= 1

  return value
```

Results & Findings

In the code above you can see that I used a few different functions. The first one was a random function in order to ensure that the dealer provides a random card. The next one was a function to define the values of each card and to add the total to the given hand when a new card is selected. There are a few more functions I defined such as one to play a round with multiple hands in case of splitting, another function to play a single round and finally a function to display player's and dealer's hands

```
while True:

    round_result, balance = play_round(balance)
```

```
    if not round_result:
        break
    play_again = input("Do you want to play another round? (Y/N)").upper()
    if play_again != 'Y':
        break
```

The code above is the main code of the program and is considered to be the main game loop. This is the code that gives players the option to play again or to leave.

Design and Implementation

I wanted to make sure that my blackjack program simulated the real experience so I made it so that the player has an initial balance and then has their option to wager and bet as much as they chose to. Once they select their bet, based on the results of the hand it will either add or deduct from their balance and then the players will get prompted to either play again or to finish the simulation.

Challenges and Solutions

The biggest challenge that I encountered was to add certain features like doubledown or split. In a true game of black jack when you select to doubledown, the player is not allowed to take another hit. Implementing this feature in my code was problematic as it kept giving the user an option to hit again. In order to fix this I decided to create a separate loop for the doubledown feature so that when the user selects it, it automatically breaks the code and the hand is done. Once I had overcome that obstacle the next problem I was faced with was the split. In traditional blackjack the user can hit again when they select to split and since I had the loop in place for the doubledown, it would also break the code for the split. To fix this I decided to create a separate function for the split and only call it in when the user selects that option. This ensured that my code won't break if put into that situation.

Discussion

The significance of this project is for everyday users to play blackjack from the comforts of their homes and not have to drive to the casino. Another critical aspect of my game is that it does not use real money. Gambling is a huge problem in the world and many people struggle with an addiction towards it. My game gives the users an option to gain the same thrill yielded from gambling but without the cost of a financial burden on their bank accounts.

Conclusion:

In conclusion, the Black Jack program developed by me successfully brings the excitement of a casino blackjack experience to the digital realm, allowing users to enjoy the game from the convenience of their homes. The implementation of key Python features, such as lists and random functions, demonstrates a thoughtful approach to replicating the unpredictability of real card games.

The design and implementation of the program showcase a commitment to authenticity, with features like betting, wagering, and balance management mirroring the dynamics of a genuine blackjack session. The inclusion of unique functions for actions like doubling down and splitting adds depth to the gaming experience, overcoming challenges in replicating these nuanced aspects of traditional blackjack.

Through thoughtful consideration of user experience, the program not only entertains but also addresses social concerns. By providing a simulated gambling experience without the involvement of real money, my Black Jack offers a responsible alternative for those seeking the thrill of the game without the financial risks associated with traditional casino gambling.

The successful resolution of challenges faced during the development process highlights the author's dedication to delivering a polished and functional product. The separate loops and functions for actions like doubling down and splitting demonstrate a meticulous problem-solving approach, ensuring that the program accurately reflects the rules of blackjack.

Beyond its entertainment value, this project holds significance in acknowledging and addressing the issue of gambling addiction. By offering a risk-free environment for users to enjoy the excitement of blackjack, the program promotes responsible gaming and aligns with a broader societal awareness of the potential dangers associated with gambling.

In essence,my Black Jack program not only provides a technically proficient and engaging gaming experience but also stands as a testament to the author's commitment to responsible and innovative game development in the ever-evolving digital landscape.

References

"Blackjack." *English*, www.hippodromecasino.com/hippodrome-casino/blackjack/#:~:text=You%20and%20fellow%20players%20are,are%20happy%20with%20your%20hand. Accessed 11 Dec. 2023.