University of St.Gallen

Python blackjack game visualised

Authors

Michel Kull
Student ID:16-728-008
michel.kull@student.unisg.ch

Gilles Custer
Student ID:16-606-758
gilles.custer@student.unisg.ch

8,789,1: Skills: Programming with Advanced Computer Languages

Dr. Mario Silic

16. May 2022

1 About the project

This program lets the user play blackjack against the dealer.

After starting the game, the user will be dealt 2 cards. He now must choose whether he wants to stick to his 2 cards (“[S]tand”) or if he wants another one (“[H]it”). Numbered cards are worth their face value, Jack/Queen/King are worth 10 and the Ace can be counted as either 1 or 11.

The goal of the game is to get as close to the number 21 as possible, without exceeding it. If the player succeeds to get closer to 21 than the dealer, the player wins. In case he exceeds 21 or the dealer gets closer to 21, the player loses. If both the player and the dealer have the same amount or both exceed 21, it is counted as a tie and no-one either wins or loses.

The game starts immediately after running the code and therefore it does not need further explanations. The code is kept as simple as possible without excluding ideas like a visual representation of the dealt cards.

2 Code

```
import random
import os

# The Card class definition
class Card:
    def __init__(self, suit, value, card_value):

        # Suit of the Card like Spades and Clubs
        self.suit = suit

        # Representing Value of the Card like A for Ace, K for King
        self.value = value

        # Score Value for the Card like 10 for King
        self.card_value = card_value

# Clear the terminal
def clear():
    os.system("clear")

# Carddesign
def print_cards(cards, hidden):

    s = ""
    for card in cards:
        s = s + "\t            "
    print(s)

    s = ""
    for card in cards:
        s = s + "\t\033[7m             \033[0m"
    print(s)

    s = ""
    for card in cards:
        if card.value == '10':
            s = s + "\t\033[7m  {}         \033[0m".format(card.value)
        else:
            s = s + "\t\033[7m  {}          \033[0m".format(card.value)
    print(s)

    s = ""
    for card in cards:
        s = s + "\t\033[7m             \033[0m"
    for i in range(2):
        print(s)

    s = ""
    for card in cards:
        s = s + "\t\033[7m      {}      \033[0m".format(card.suit)
    print(s)
```

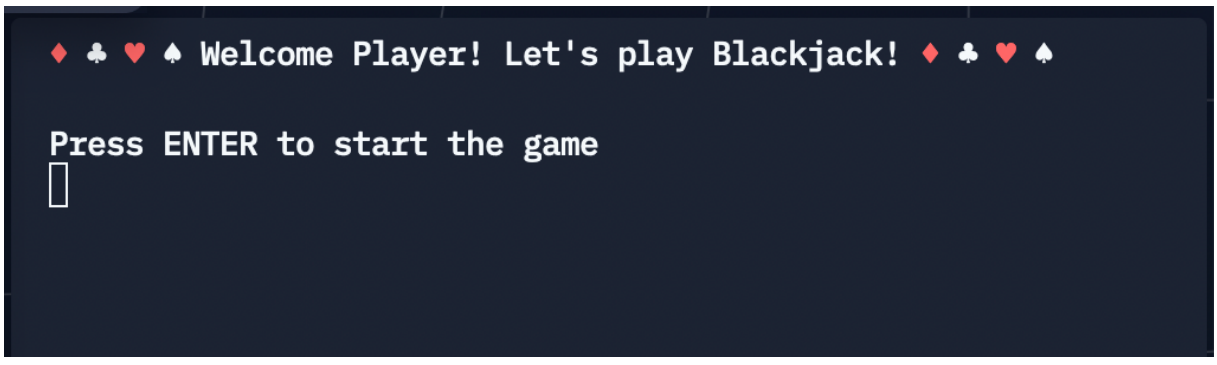
```
    s = ""
    for card in cards:
        s = s + "\t\033[7m             \033[0m"
    for i in range(2):
        print(s)

    s = ""
    for card in cards:
        if card.value == '10':
            s = s + "\t\033[7m         {}  \033[0m".format(card.value)
        else:
            s = s + "\t\033[7m         {}   \033[0m".format(card.value)
    print(s)

    s = ""
    for card in cards:
        s = s + "\t\033[7m             \033[0m"
    print(s)

    print()

print("\033[31m♦\033[0m", "♣", "\033[31m♥\033[0m", "♠", "Welcome Player! Let's play Blackjack!",
"\033[31m♦\033[0m", "♣", "\033[31m♥\033[0m", "♠")
print(" ") #use one empty line to seperate the 2 prints
print("Press ENTER to start the game")
input()
  # Function for a game of blackjack

def blackjack_game(deck):

  # Define the deck at beginning of function to reset the deck for each round
    deck = []

    # Loop for every type of suit
    for suit in suits:

        # Loop for every type of card in a suit
        for card in cards:

            # Adding card to the deck
            deck.append(Card(suits_values[suit], card, cards_values[card]))

    # Cards for both dealer and player
    player_cards = []
    dealer_cards = []

    # Scores for both dealer and player
    player_score = 0
    dealer_score = 0

    clear()

    # Initial dealing for player and dealer

    while len(player_cards) < 2:

        # Randomly dealing a card
        player_card = random.choice(deck) #Randomly dealing a card to the player
        player_cards.append(player_card) #Add dealt card to players list of cards
        deck.remove(player_card) #Remove the dealt card from the original deck

        # Updating the player score
        player_score += player_card.card_value

        # In case both the cards are Ace, make the first ace value as 1
        if len(player_cards) == 2:
            if player_cards[0].card_value == 11 and player_cards[1].card_value == 11:
                player_cards[0].card_value = 1
                player_score -= 10

        # Print player cards and score
        print("Player cards: ")
        print("Value = ", player_score)
        print_cards(player_cards, False)


        # Randomly dealing a card
        dealer_card = random.choice(deck) #Randomly dealing a card to the dealer
        dealer_cards.append(dealer_card) #Add dealt card to dealers list of cards
        deck.remove(dealer_card) #Remove the dealt card from the original deck
```

```
    # Updating the dealer score
    dealer_score += dealer_card.card_value

    # Print dealer card and score
    print("Dealer cards: ")
    if len(dealer_cards) == 1:
        print("Value = ", dealer_score)
        print_cards(dealer_cards, False)

    else:
        print("Value = ", dealer_score - dealer_cards[-1].card_value)
        print_cards(dealer_cards[:-1], True)



    # In case both the cards are Ace, make the second ace value as 1
    if len(dealer_cards) == 2:
        if dealer_cards[0].card_value == 11 and dealer_cards[1].card_value == 11:
            dealer_cards[1].card_value = 1
            dealer_score -= 10

# Player gets a blackjack
if player_score == 21:
    print("Player has a Blackjack!")
    print("Player has won!")
clear()

# Print dealer and player cards
print("Dealer cards: ")
print("Value = ", dealer_score - dealer_cards[-1].card_value)
print_cards(dealer_cards[:-1], True)


print()

print("Player cards: ")
print("Value = ", player_score)
print_cards(player_cards, False)


# Managing the player moves
while player_score < 21:
    choice = input("Enter H to Hit or S to Stand : ")

    # Sanity checks for player's choice
    if len(choice) != 1 or (choice.upper() != 'H' and choice.upper() != 'S'):
        print("Wrong choice! Please enter H or S.")

    # If player decides to HIT
    if choice.upper() == 'H':

        # Dealing a new card
        player_card = random.choice(deck)
        player_cards.append(player_card)
        deck.remove(player_card)

        # Updating player score
        player_score += player_card.card_value

        # Updating player score in case player's card have ace in them
        c = 0
        while player_score > 21 and c < len(player_cards):
            if player_cards[c].card_value == 11:
                player_cards[c].card_value = 1
                player_score -= 10
                c += 1
            else:
                c += 1

        clear()

        # Print player and dealer cards
        print("Dealer cards: ")
        print("Value = ", dealer_score - dealer_cards[-1].card_value)
        print_cards(dealer_cards[:-1], True)


        print()

        print("Player cars: ")
```

```
            print("Value = ", player_score)
            print_cards(player_cards, False)


        # If player decides to Stand
        if choice.upper() == 'S':
            break


clear()

# Print player and dealer cards
print("Player cards: ")
print("Value = ", player_score)
print_cards(player_cards, False)


print()
print("Dealer pulls another card....")

print("Dealer cards: ")
print("Value = ", dealer_score)
print_cards(dealer_cards, False)


# Managing the dealer moves
while dealer_score < 17:
    clear()

    print("Dealer decides to hit.....")

    # Dealing card for dealer
    dealer_card = random.choice(deck)
    dealer_cards.append(dealer_card)
    deck.remove(dealer_card)

    # Updating the dealer's score
    dealer_score += dealer_card.card_value

    # Updating player score in case player's card have ace in them
    c = 0
    while dealer_score > 21 and c < len(dealer_cards):
        if dealer_cards[c].card_value == 11:
            dealer_cards[c].card_value = 1
            dealer_score -= 10
            c += 1
        else:
            c += 1

    # print player and dealer cards
    print("Player cards: ")
    print("Value = ", player_score)
    print_cards(player_cards, False)


    print()

    print("Dealer cards: ")
    print("Value = ", dealer_score)
    print_cards(dealer_cards, False)


# Dealer busts
if dealer_score > 21:
    print("Dealer busted! You win!")
# Player busts
if player_score > 21:
    print("Player busted!")

# Dealer gets a blackjack
if dealer_score == 21:
    print("Dealer has a Blackjack! Player loses")

# TIE Game
if dealer_score == player_score:
    print("Tie Game!")

# Check if player has a Blackjack
if player_score == 21:
    print("Player has a Blackjack!")
```

```
    # Player Wins
    if player_score > dealer_score and player_score <= 21:
        print("Player wins!")

    # Dealer Wins
    if dealer_score > player_score and dealer_score <=21:
        print("Dealer wins!")

if __name__ == '__main__':

    # The type of suit
    suits = ["Spades", "Hearts", "Clubs", "Diamonds"]

    # The suit value as unicode symbols
    suits_values = {"Spades":"\u2660", "Hearts":"\033[41m\u2764", "Clubs": "\u2663", "Diamonds":
"\033[41m\u2666"}

    # The type of card
    cards = ["A", "2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K"]

    # The card value
    cards_values = {"A": 11, "2":2, "3":3, "4":4, "5":5, "6":6, "7":7, "8":8, "9":9, "10":10,
"J":10, "Q":10, "K":10}

    # The deck of cards
    deck = []

    # Loop for every type of suit
    for suit in suits:

        # Loop for every type of card in a suit
        for card in cards:

            # Adding card to the deck
            deck.append(Card(suits_values[suit], card, cards_values[card]))

# Let the player decide if he wants to keep playing
a = True
while a:
    blackjack_game(deck)
    print("Do you want to keep playing? Press ENTER to play again or press X to exit.")
    b = input()
    if b.lower() == "x":
        a = False
        print("\033[31m♦\033[0m", "♣", "\033[31m♥\033[0m", "♠", "Thank you for playing with us! See
you next time.", "\033[31m♦\033[0m", "♣", "\033[31m♥\033[0m", "♠")
    else:
        a = True
```

3 Example: input and output



```
Dealer cards:
Value =  9

 9
    ♣
      9


Player cards:
Value =  20

 J          K
    ♠          ♠
      J          K

Enter H to Hit or S to Stand : S
```

```
Dealer pulls another card....
Dealer cards:
Value =  17

 9          8
    ♣          ♣
      9          8
```

```
Player wins!
Do you want to keep playing? Press ENTER to play again or p
ress X to exit.
X
♦ ♣ ♥ ♠ Thank you for playing with us! See you next time. ♦
 ♣ ♥ ♠
```