Day - 11

**The Blackjack Capstone Project**

Rules and Code for Blackjack card Game

**11.1 Extend and "+="**

***List.extend(obj)*** is similar to ***List += obj***. Here ***obj*** must be the ***list*** type. i.e. we can marge/append a ***List*** to a ***List***. To add a new element we need to use ***append()*** function.

|  |  |
| --- | --- |
| ERROR | Fixes the ERROR |
| **for** i **in** **range**(2):      new\_card = **deal\_card**()      user\_cards += new\_card  Here error is thrown. ***new\_card*** is ***int*** type but ***user\_cards*** is a ***list***. | **for** i **in** **range**(2):      new\_card = **deal\_card**()      user\_cards += [new\_card]  Notice square braces ***[]***,***[new\_card]***. Now '***+=***' works, no error thrown. Because ***[new\_card]*** is a ***list*** itselfhence no type-error will arise. |

* **extend & += ,:** extend a list by appending, it has to be a ***'list'*** itself. ***append()*** is used when you want to add a single item, not a ***list***.

sum(**iterable**: Iterable[\_T], /) -> Union[\_T, int]

param iterable: Iterable[\_T]

Return the sum of a 'start' value (default: 0) plus an iterable of numbers. When the iterable is empty, return the start value. This function is intended specifically for use with numeric values and may ***reject non-numeric types***.

**11.2 Practice Version**

# # # # # # # # # # # # # # Our Blackjack House Rules # # # # # # # # # # # # # # # # #

The deck is unlimited in size.

There are no jokers.

The Jack/Queen/King all count as 10.

The the Ace can count as 11 or 1.

Use the following list as the deck of cards:

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

The cards in the list have equal probability of being drawn.

Cards are not removed from the deck as they are drawn.

# # # # # # # # # # # # # # # # # # # # Hints # # # # # # # # # # # # # # # # # # # # #

* The dealer gives the other person one card up and one card down, then the same for themselves. Now they look at their bottom card and add the face value to the top card. If you are about ten or less from twenty-one then you want to hit(get another card), If you are close to twenty-one you want to stay(keep the cards you have). If you break(go over) 21 then you have busted, this means that you are outfor the rest of this hand.
* Hit definition- when you want to get another card to get closer to 21, you can hit as many times as you want without going over 21.
* Stay definition- when you want to keep the same cards you have because you are confident you will beat your opponents hand, or you will bust if you hit.
* Bust definition- when the sum of your cards value is over 21 and you have to stop for that hand, and your opponent wins the hand because you bust.

FACE VALUES:

Ace-either one or eleven(the players choice)

Two-2

three-3

Four-4

Five-5

Six-6

Seven-7

Eight-8

Nine-9

Ten-10

Jack-10

Queen-10

King-10

Practice

art\_blacK\_Jack.py

logo = """

.------.            \_     \_            \_    \_            \_

|A\_  \_ |.          | |   | |          | |  (\_)          | |

|( \/ ).-----.     | |\_\_ | | \_\_ \_  \_\_\_| | \_\_\_  \_\_ \_  \_\_\_| | \_\_

| \  /|K /\  |     | '\_ \| |/ \_` |/ \_\_| |/ / |/ \_` |/ \_\_| |/ /

|  \/ | /  \ |     | |\_) | | (\_| | (\_\_|   <| | (\_| | (\_\_|   <

`-----| \  / |     |\_.\_\_/|\_|\\_\_,\_|\\_\_\_|\_|\\_\ |\\_\_,\_|\\_\_\_|\_|\\_\\

      |  \/ K|                            \_/ |

      `------'                           |\_\_/

"""

**import** art\_blacK\_Jack

**import** random

**print**(art\_blacK\_Jack**.**logo)

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

**def** **draw\_card**():

**return** random**.choice**(cards)

#*Extreme case*

#*plyr\_card = [11, 11]*

#*pc\_card = [11, 11]*

plyr\_card = [**draw\_card**(), **draw\_card**()]

pc\_card = [**draw\_card**(), **draw\_card**()]

**def** **total**(plr\_crd):

    plr\_total = 0

**for** crd **in** plr\_crd:

        plr\_total += crd

**if** (11 **in** plr\_crd):

**if** plr\_total **>** 21:

**print**(f"index of 11 : {plr\_crd**.index**(11)}")

            plr\_crd[plr\_crd**.index**(11)] = 1

            plr\_total -= 10

**print**(f"Place of 11 , replaced by 1 now : {plr\_crd}")

**return** plr\_total

**print**(f"player cards : {plyr\_card} and total = {**total**(plr\_crd = plyr\_card)}")

**print**(f"pC cards : {pc\_card} and total = {**total**(plr\_crd = pc\_card)}")

**def** **pc\_dicision**():

**if** (**total**(pc\_card) **<** 21) **and** (11 **not** **in** pc\_card):

        dCsn = random**.choice**(["S", "H"])

**if** dCsn **==** "H":

            hit2 = **draw\_card**()

            pc\_card**.append**(hit2)

**elif** (**total**(pc\_card) **<** 21) **and** (11 **in** pc\_card):

**if**  **total**(pc\_card) **>** 21:

**print**(f"index of 11 : {pc\_card**.index**(11)}")

            pc\_card[pc\_card**.index**(11)] = 1

**print**(f"Place of 11 , replaced by 1 now : {pc\_card}")

        dCsn = random**.choice**(["S", "H"])

**if** dCsn **==** "H":

            hit2 = **draw\_card**()

            pc\_card**.append**(hit2)

**def** **ask**():

    aSk = **input**("Want \"Hit\" or \"Stay\" or \"Call\"? Input 'H' or 'S' 'C'")

**if** aSk **==** "H":

        hit = **draw\_card**()

        plyr\_card**.append**(hit)

**print**(plyr\_card)

**pc\_dicision**()

**ask**()

**elif** aSk **==** "S":

**pc\_dicision**()

**ask**()

**elif** aSk **==** "C":

**print**("Game over")

**print**(f"player cards : {plyr\_card} and total = {**total**(plr\_crd = plyr\_card)}")

**print**(f"pC cards : {pc\_card} and total = {**total**(plr\_crd = pc\_card)}")

**if** **total**(pc\_card) **==** **total**(plyr\_card):

**print**("Draw")

**elif** (**total**(pc\_card) **<** **total**(plyr\_card) **<=** 21):

**print**("You win")

**elif** **total**(plyr\_card) **>** 21 **>=** **total**(pc\_card):

**print**("You are Busted. PC wins ")

**elif** **total**(plyr\_card) **<** **total**(pc\_card) **<=** 21:

**print**("PC wins ")

**elif** (**total**(pc\_card) **>** 21) **and**  (**total**(plyr\_card) **>** 21):

**print**("Draw")

**elif** (**total**(plyr\_card) **<=** 21 **<** **total**(pc\_card)):

**print**("You win")

**ask**()

#*python blacK\_Jack.py*

Instructor version

**11.3 Instructor version**

#*# # # # # # # # # # # # # #  Our Blackjack House Rules # # # # # # # # # # # # # # # # #*

#*The deck is unlimited in size.*

#*There are no jokers.*

#*The Jack/Queen/King all count as 10.*

#*The the Ace can count as 11 or 1.*

#*Use the following list as the deck of cards:*

#*cards = [11, 2, 3, 4, 5, 6, 7, 8, 9, 10, 10, 10, 10]*

#*The cards in the list have equal probability of being drawn.*

#*Cards are not removed from the deck as they are drawn.*

#*# # # # # # # # # # # # # # # # # # # #  Hints # # # # # # # # # # # # # # # # # # # # #*

**import** art\_blacK\_Jack

**import** random

**import** os

#*---------------- Randomly deal cards*

**def** **deal\_card**():

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

    card = random**.choice**(cards)

**return** card

#*---------------- Create a function called compare() and pass in the user\_score and computer\_score:*

    #*If the computer and user both have the same score, then it's a draw.*

    #*If the computer has a blackjack (0), then the user loses.*

    #*If the user has a blackjack (0), then the user wins.*

    #*If the user\_score is over 21, then the user loses.*

    #*If the computer\_score is over 21, then the computer loses.*

    #*If none of the above, then the player with the highest score wins.*

**def** **compare**(u\_sCr, pc\_ScR):

**if** u\_sCr **==** pc\_ScR:

**return** f"Draw"

**elif** u\_sCr **==** 0:

**return** f"User wins"

**elif** (pc\_ScR **==** 0):

**return** f"Pc wins"

**elif** pc\_ScR **>** 21:

**return** f"User wins !!! PC busted"

**elif** u\_sCr **>** 21:

**return** f"Pc wins !!! User busted"

**elif** pc\_ScR **>** u\_sCr:

**return** f"Pc wins"

**elif** pc\_ScR **<** u\_sCr:

**return** f"You win !!"

#*---------------- Calculate score:*

    #*Inside calculate\_score() check for a blackjack (a hand with only 2 cards: ace + 10) and return 0 instead of the actual score. 0 will represent a blackjack in our game.*

    #*Inside calculate\_score() check for an 11 (ace). If the score is already over 21, remove the 11 and replace it with a 1. You might need to look up append() and remove().*

**def** **calculate\_score**(cRDs):

    """ Take a list of cards and return the score calculated from the cards """

**if** (**sum**(cRDs) **==** 21) **and** (**len**(cRDs) **==** 2):

**return** 0

**if** (**sum**(cRDs) **>** 21) **and** (11 **in** cRDs):

        cRDs**.append**(1)

        cRDs**.remove**(11)

**return** **sum**(cRDs)

#*---------------- The main function*

**def** **blackjack\_main**():

**print**(art\_blacK\_Jack**.**logo)

    #*DEal the cards: Deal the user and computer 2 cards each using deal\_card() and append().*

    user\_cards = []

    pc\_cards = []

**for** i **in** **range**(2):

        user\_cards**.append**(**deal\_card**())

        pc\_cards**.append**(**deal\_card**())

    #*Call calculate\_score(). If the computer or the user has a blackjack (0) or if the user's score is over 21, then the game ends*

    #*If the game has not ended, ask the user if they want to draw another card. If yes, then use the deal\_card() function to add another card to the user\_cards List. If no, then the game has ended.*

    is\_game\_over = **False**

**while** **not** is\_game\_over:

        user\_score = **calculate\_score**(user\_cards)

        pc\_score = **calculate\_score**(pc\_cards)

**print**(f"Your cards : {user\_cards} and total = {user\_score}")

**print**(f"Pc's first card : {pc\_cards[0]} ")

**if** (user\_score **==** 0) **or** (pc\_score **==** 0) **or** (user\_score **>** 21):

            is\_game\_over = **True**

**else**:

            user\_should\_deal = **input**("Type 'y' to get another card, type 'n' to pass: ")

**if** user\_should\_deal **==** 'y':

                user\_cards**.append**(**deal\_card**())

**else**:

                is\_game\_over = **True**

    #*PC choice: Once the user is done, it's time to let the computer play. The computer should keep drawing cards as long as it has a score less than 17.*

**while** 0 **<** pc\_score **<** 17 :

        pc\_cards**.append**(**deal\_card**())

        pc\_score = **calculate\_score**(pc\_cards)

**print**(f"Your final hand : {user\_cards} and Final Score = {user\_score}")

**print**(f"PC's final hand : {pc\_cards} and Final Score = {pc\_score}")

**print**(**compare**(user\_score, pc\_score))

#*Ask the user if they want to restart the game. If they answer yes, clear the console and start a new game of blackjack and show the logo from art.py.*

**while** **input**(f"Wanna paly again? Press y if yes: ") **==** 'y':

    os**.system**("cls")

**blackjack\_main**()

#*python blacK\_Jack\_angela.py*