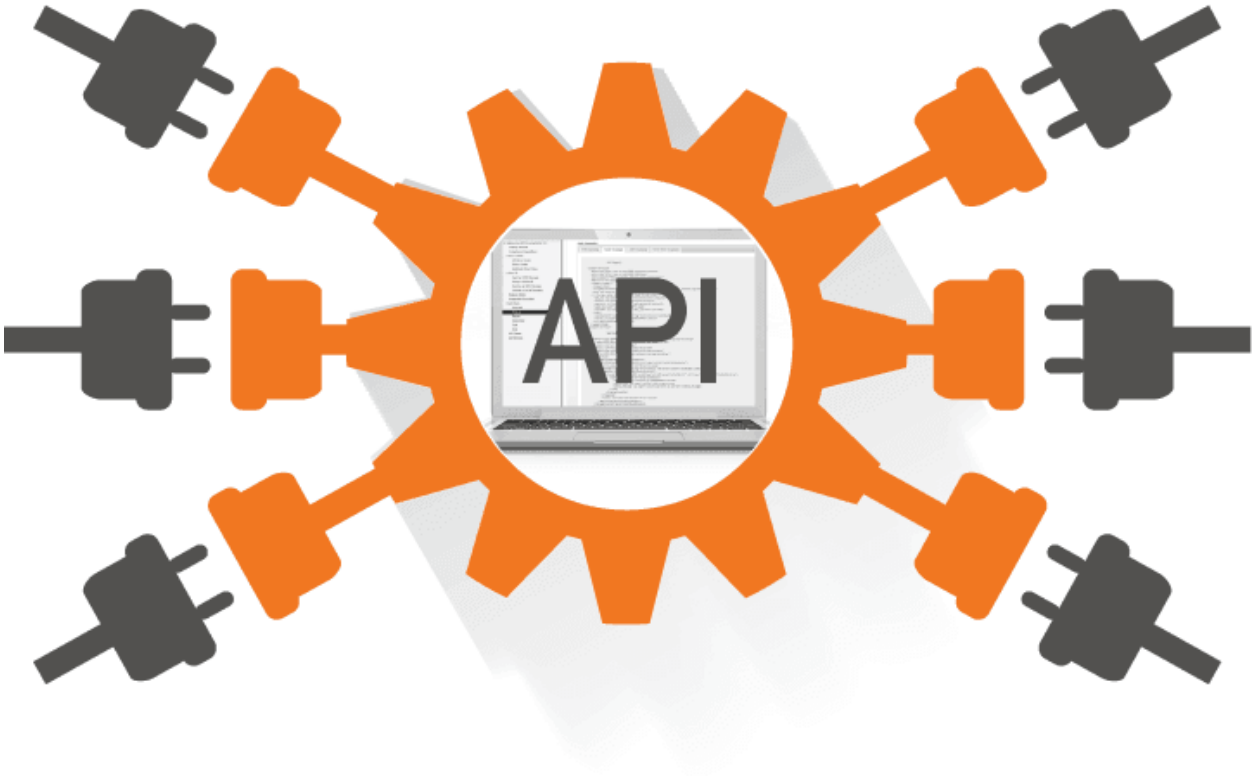
APIS

Ian Nogueira



INTRODUCCIÓN

En este proyecto dividiremos la explicación en dos apartados, el de script normal y el de textual. En estos dos apartados explicaremos como se ha hecho el script, como funciona, ejemplos del funcionamiento y demás. En este proyecto he decidido usar de api la ***Deck of Cards****(https://deckofcardsapi.com/). Esta api se basa en poder usar mazos, cartas y pilas a tu antojo para hacer diversos juegos o simplemente para hacer que aparezca una carta por pantalla. Tiene varias funciones como generar una nueva baraja, sacar una carta o barajar la baraja. Todo esto funciona con un id único de la baraja, el cual se genera automáticamente cuando usas la opción de generar nueva baraja. Para darle un uso más práctico a esta api, en vez de hacer que aparezca una imagen de una carta sin más, he implementado el juego Blackjack en conjunto con la api. Este juego es una versión simple sin tener en cuenta el factor de apostar para jugar. Luego está el apartado del textual, en el cual se explicará cómo he utilizado la herramienta de textual de manera muy básica para escribir mensajes y luego como la he implementado en el juego.*

SCRIPT NORMAL

El script está dividido en funciones que hacen todo lo necesario para que el juego funcione. Luego, está la parte principal donde se llaman a todas las funciones necesarias para generar el loop del juego. Comenzaremos con la explicación de los imports.

IMPORTS

```
import time
import json
import requests
import sys
from fabulous import utils,image
```

Los imports son para poder hacer que python pueda leer las requests que da la api y para que pueda funcionar correctamente con archivos json. El resto son para hacer pausas como el time o para que la imagen de la carta se imprima por la pantalla.

FUNCIONES

```
def new_deck():
    api_url="https://deckofcardsapi.com/api/deck/new/"

    response = requests.get(api_url)

    info = response.json()

    deck_id=info["deck_id"]
```

```
    return deck_id
```

Esta función lo que hace es generar una nueva baraja, sacar el id de dicha baraja y luego sacarla en el return para que pueda ser usada más tarde.

```
def draw_card(deck_id, cards):
    api_url=
"https://deckofcardsapi.com/api/deck/"+deck_id+"/draw/?coun
t="+str(cards)

    response = requests.get(api_url)

    info = response.json()

    card=info["cards"][0]["image"]
    card_value=info["cards"][0]["value"]
    if   card_value=="KING"   or   card_value=="QUEEN"   or
card_value=="JACK":
        card_value=10

    if card_value=="ACE":
        value=input("Que valor quieres que tenga tu as, 1
o 11?")
        if str(value)=="11":
            card_value=11
        elif str(value)=="1":
            card_value=1
        else:
            print("Valor incorrecto, te pondremos un 1")

    player_deck.append(card_value)
```

```
    images= requests.get(card).content

    image_name="carta.png"
with open(image_name, "wb") as handler:
        handler.write(images)
        img = image.Image(image_name)
        time.sleep(1)
        print(img)

    if cards > 1:
        card2=info["cards"][1]["image"]
        card_value2=info["cards"][1]["value"]
        if card_value2=="KING" or card_value2=="QUEEN" or
card_value2=="JACK":
            card_value2=10

        if card_value2=="ACE":
            value2=input("Te ha tocado un as, que valor
quieres que tenga? ¿1 o 11?")

            if str(value2)=="11":
                card_value2=11
            elif str(value2)=="1":
                card_value2=1
            else:
                print("Valor  incorrecto,  te  pondremos
un 1")
                card_value2=1
        player_deck.append(card_value2)

        images2= requests.get(card2).content
```

```
            image_name2="carta2.png"
            with open(image_name2, "wb") as handler:
                handler.write(images2)
                img2 = image.Image(image_name2)
                time.sleep(1)
                print(img2)
```

Esta función lo que hace es sacar una carta del mazo. Para hacer esto le mandamos a la función la id del mazo que hemos creado antes, así usaremos todo el rato el mismo mazo. Luego, sacaremos del json que recibimos dos cosas, la imagen de la carta, y el valor de la carta. Sacamos la imagen para luego printarla por pantalla y así saber que carta le toca a cada jugador. El valor antes de hacer nada con ella, miraremos si es una figura o si es un as. En el caso de que sea una figura, cambiaremos el valor por un 10, y en el caso de que toque un as, haremos la pregunta de si el jugador quiere que ese as sea un 1 o un 11, si pone otro número se asignará un 1 automáticamente. Luego de reasignar estos valores se añadirá a la lista llamada player_deck y printara la imagen de la carta por la terminal. Luego de hacer esto, se mirara si el valor de cards es mayor de uno, si es así significa que hay más de una carta que sacar, por lo que sí es más de uno, se volverá a hacer el proceso de antes pero con la segunda carta.

```
def croupier_draw_card(deck_id, cards):
    api_url=
"https://deckofcardsapi.com/api/deck/"+deck_id+"/draw/?coun
t="+str(cards)
    response = requests.get(api_url)
    info = response.json()
```

```
    card=info["cards"][0]["image"]
    card_value=info["cards"][0]["value"]

    if   card_value=="KING"   or   card_value=="QUEEN"   or
card_value=="JACK":
         card_value=10
    if card_value=="ACE":
         card_value=11
    croupier_deck.append(card_value)

    images= requests.get(card).content
    image_name="croupier_card.png"

    with open(image_name, "wb") as handler:
         handler.write(images)
         img = image.Image(image_name)
         time.sleep(1)
         print(img)

    if cards > 1:
         card2=info["cards"][1]["image"]
         card_value2=info["cards"][1]["value"]

         if card_value2=="KING" or card_value2=="QUEEN" or
card_value2=="JACK":
              card_value2=10
         if card_value2=="ACE":
              if card_value==11:
                   card_value2=1
              card_value2=11
         croupier_deck.append(card_value2)
```

```
            images2= requests.get(card2).content
            image_name2="croupier_card2.png"

            with open(image_name2, "wb") as handler:
                handler.write(images2)
                img2 = image.Image(image_name2)
                time.sleep(1)
                print(img2)
```

Esta función hace lo mismo que la función de draw_card pero con el croupier. Las diferencias son que cuando toca un as lo que hará será transformar su valor automáticamente a 11 y si le ha tocado un as anteriormente el as será un 1, por lo que nunca podrá sumar dos 11. Otra diferencia es que la lista que cambia es la de croupier_deck en vez de player_deck.

```
def shuffle_cards(deck_id):

api_url="https://deckofcardsapi.com/api/deck/"+deck_id+"/shuffle/"

    response = requests.get(api_url)

    info = response.json()
```

Esta función lo que hace es mezclar la baraja que hemos creado al principio, ya que cuando creas una baraja está ordenada por números y por palos, por lo que si sacamos dos cartas, tocará el as de picas y el dos de picas todo el rato.

```
def count(sum_num, sum_num2, final_count):

```

```
    for num in player_deck:
        sum_num += int(num)
    print("Esta es tu suma: "+str(sum_num))
    print("")

    for num in croupier_deck:
        sum_num2 += int(num)

    if int(sum_num2)>=16:
        print("Esta es la suma del croupier: "+str(sum_num2))
        print("")
        print("El croupier se planta")
        print("")
        if final_count==True:
            victory(sum_num, sum_num2)
        return "stop"
    else:
        print("Esta es la suma del croupier: "+str(sum_num2))
        print("")
    if final_count==True:
        victory(sum_num, sum_num2)
```

Esta función lo que hará será sumar las cartas del player y las del croupier. Cuando cuenta las cartas del croupier tendrá en cuenta si la suma del croupier es más de 16 o si el valor de final_count, que es un valor que determinará si se lanza la función de victory. Si el número es mayor de 16 comprobará la función de final_count y luego devolverá un stop, esto hará que

el croupier pare de pedir cartas ya que una de las normas es que no puede pedir cartas a partir de 16.

```
def victory(sum_num, sum_num2):
    if sum_num>21:
        print("Has perdido, el croupier gana")
    elif sum_num2>21:
        print("El croupier ha perdido, tu ganas")
    elif sum_num>sum_num2:
        print("Has ganado, el croupier pierde")
    elif sum_num<sum_num2:
        print("El croupier ha ganado, tu pierdes")
    elif sum_num==sum_num2:
        print("Empate")
    for i in range(len(player_deck)):
        player_deck.pop()
    for i in range(len(croupier_deck)):
        croupier_deck.pop()
```

Esta función lo que hará será comprobar las manos del jugador y del croupier y hará comprobaciones para saber quién ha ganado de los dos. Luego de hacer la comprobación para determinar la victoria limpiará las dos manos, la del jugador y la del croupier.

```
def rules():
    print("1-El objetivo del juego es llegar a 21, quien mas cerca esté, gana.")
    print("2-A la hora de sumar las cartas, se suman por su número excepto el as y las figuras:")
```

```
    print("El AS puede contar como un 1 o como un 11 dependiendo de lo que decida el jugador, las figuras cuentan todas 10.")
    print("3-El croupier se plantara en el momento que alcance la puntuación de 16 o más.")
    print("4-Cuando al croupier le toca un as, siempre será 11 a no ser que se pase de 21, entonces será 1.")
    print("5-Puedes pedir las cartas que quieras, cuando llegues a 22 o más habrás perdido.")
```

Esta función lo que hará será pintar todas las normas que tiene este juego.

```
def game():
    opc=False

    while not opc:
        print("Pues comencemos")
        print(deck_id)
        print("")
        shuffle_cards(deck_id)
        print("Estas son tus cartas")
        draw_card(deck_id,2)
        print(player_deck)
        print("Estas son las cartas del croupier")
        croupier_draw_card(deck_id,2)
        print("Tus cartas"+str(player_deck))
        print("Cartas del croupier"+str(croupier_deck))
        print("")
        count(sum_num,sum_num2, False)
        opcion=  input("Quieres sacar otra carta?(Y/n)")
```

```
            while opcion!="n":
                if opcion.lower()=="y":
                    draw_card(deck_id,1)
                    print(player_deck)
                else:
                    break
                opcion=     input("Quieres     sacar     otra
carta?(Y/n)" )
            ok=count(sum_num, sum_num2, False)
            while ok!= "stop":
                if ok=="stop":
                    break
                else:
                    print("El croupier saca otra carta")
                    print("")
                    croupier_draw_card(deck_id, 1)
                    print(croupier_deck)
                ok=count(sum_num, sum_num2, False)
            time.sleep(1)
            print("RECUENTO FINAL")
            print("--------------")
            count(sum_num, sum_num2, True)
            print("")
            break
```

Está función es la principal ya que es el juego como tal. Primero entra en el bucle para que no pare el juego. Luego printa las cartas del jugador y las del croupier. Luego preguntará primero si el jugador quiere sacar otra carta, dependiendo de lo que diga sacara otra o no, si la saca no parara de preguntar si quiere sacar otra carta hasta que el jugador diga que no. Cuando diga que no el croupier no parara de

sacar cartas hasta que la suma de sus cartas sea mayor de 16. Cuando pare de sacar cartas hará un último conteo donde decidirá quién gana. Cuando se decide quien gana vuelve a comenzar el juego si el jugador quiere.

PROGRAMA

Ahora explicaré lo que queda fuera de las funciones y los imports.

```
deck_id=new_deck()
sum_num=0
sum_num2=0
player_deck=[]
croupier_deck=[]
print("Bienvenido a la partida de blackjack")
salida=""
```

Aquí se declaran las variables globales que usará todo el programa como la player_deck y la croupier_deck o el id de la baraja. También se declarara la salida del bucle del juego y el print de bienvenida al juego.

```
while salida!="s":
    opc=    input("Quieres    comenzar    la    partida    o
salir?(Y/s)")
    if opc.lower() == "s":
        print("Esta bien, adios")
        break
    opc2=input("Quieres ver las reglas?(Y/n)")
    if opc2.lower()=="y":
        rules()
```

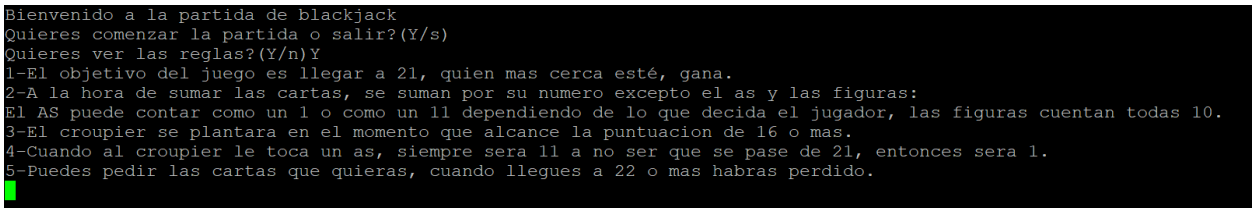
```
            time.sleep(20)
            print("")
        else:
            print("Si no quieres leer las reglas, el juego comienza")
        game()
```

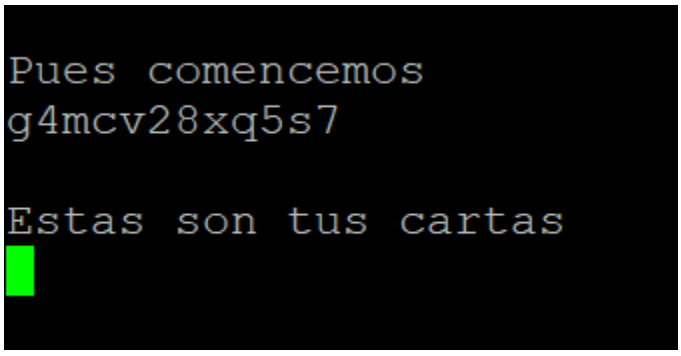
Esta parte es donde empezaremos el juego y donde daremos las opciones de comenzar o las de mostrar las reglas. Luego haremos saltar la función del juego principal para comenzar.

JUEGO

Ahora haré una demostración del juego con capturas para demostrar su funcionalidad.

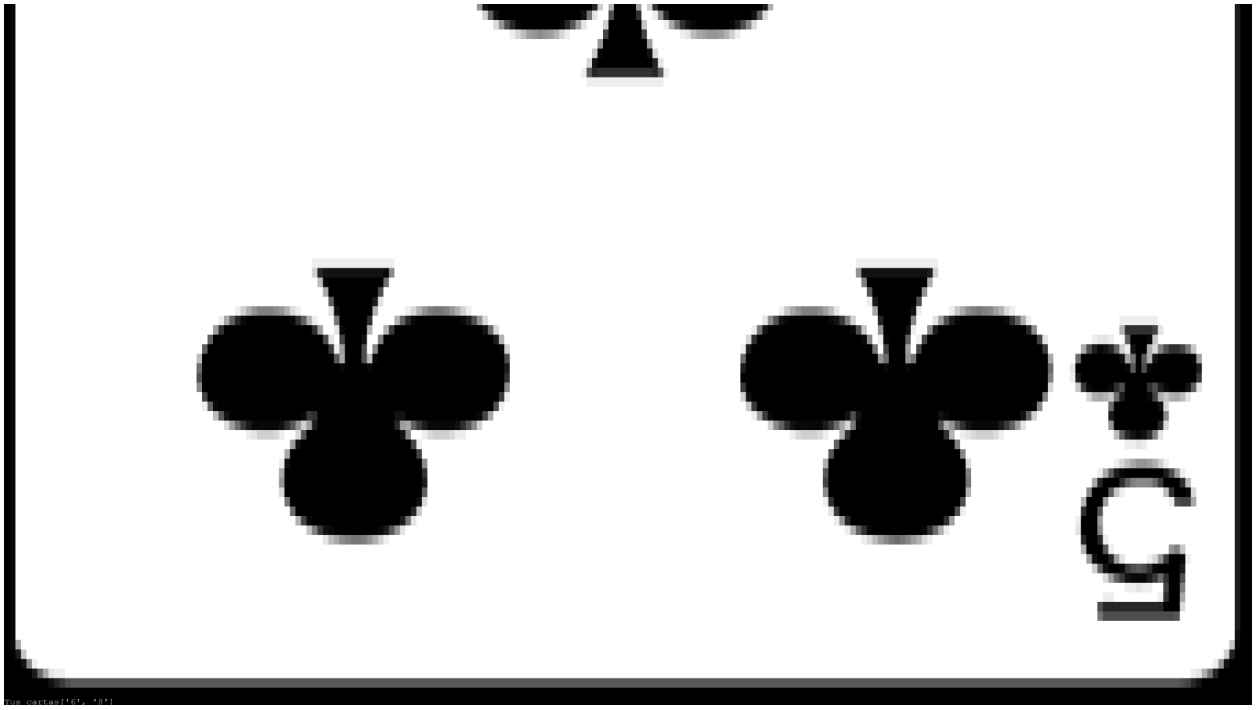
Inicio:





Como podemos ver, al inicio nos dará la bienvenida y nos preguntará si queremos ver las reglas. Cuando le decimos que sí nos dará 25 segundos para leerlas y luego saltará al juego, donde nos mostrará el id de la baraja y el mensaje de nuestras cartas.

Juego:

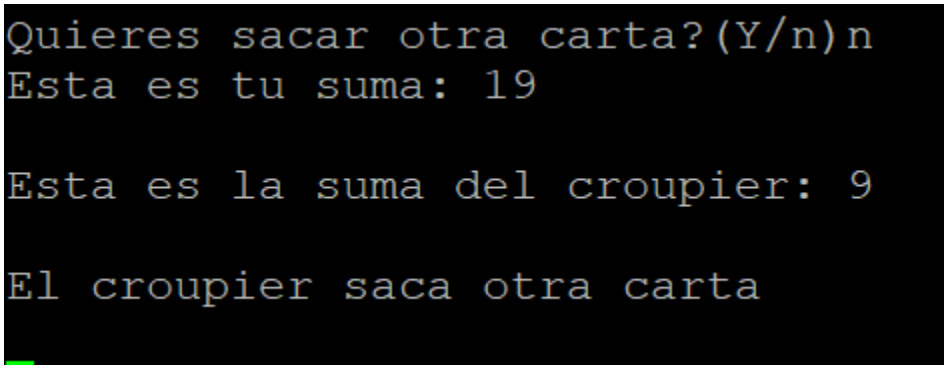


```
Tus cartas['6', '8']
Cartas del croupier['4', '5']

Esta es tu suma: 14

Esta es la suma del croupier: 9

Quieres sacar otra carta?(Y/n)
```

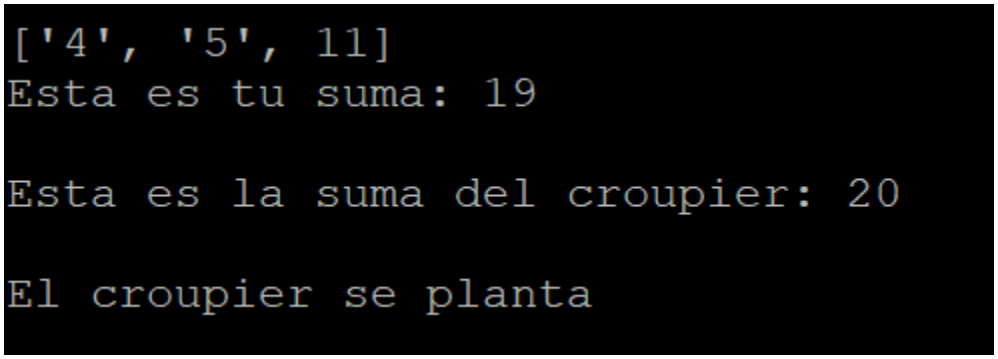
Como podemos ver en la terminal se muestra la carta en una resolución muy alta. Luego de mostrar todas las cartas nos

mostrará cual es el valor de las cartas que nos ha tocado a nosotros y su suma y lo mismo para el croupier. Luego nos preguntará si queremos sacar otra carta.

Nos mostrará otra carta y luego nos preguntará si queremos sacar otra carta, si le decimos que no pasará el turno al croupier.

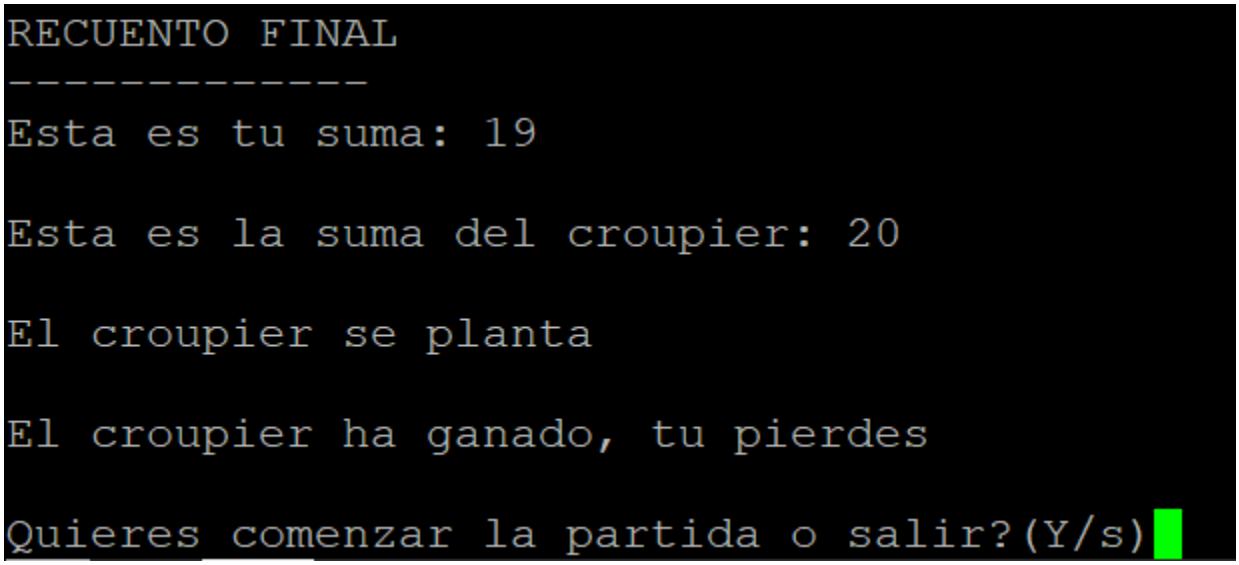


Al pasarle el turno al croupier nos saldrá la suma de nuestras cartas y las suyas y el mensaje de que el croupier sacará otra carta ya que el número es menor a 16.

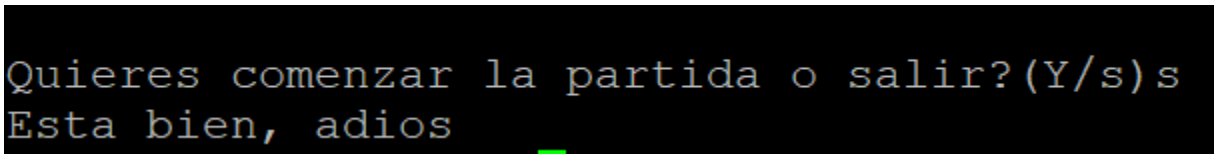


Aquí saldrá otra imagen de la carta que le ha tocado al croupier y toda su mano. Luego nos mostrará nuestra suma y la suya. En este caso muestra que el croupier se planta ya que su suma ha llegado a más de 16. Si este no hubiera sido el caso, habría vuelto a sacar otra carta.

Final:



En la parte final saldrá todo el recuento final junto al ganador, en este caso el ganador es el croupier ya que su suma se ha acercado más a 21 que la nuestra. Luego nos preguntará si queremos comenzar la partida para empezar otra vez.



Si le decimos que salir (s), nos dirá que adios y pararemos el juego.

TEXTUAL

Primero lo que he hecho ha sido seguir el tutorial para ver como funcionaba. Las líneas de código son las siguientes.

```
from textual.app import App, ComposeResult
from textual.widgets import Button, Label
```

```
class HelloWorld(App):
    CSS_PATH = "hello.css"
    def compose(self) -> ComposeResult:
        self.close_button = Button("Close", id="close")
        yield Label("Hello Textual", id="hello")
        yield self.close_button
    def on_mount(self) -> None:
        self.screen.styles.background = "darkblue"
        self.close_button.styles.background = "red"
     def on_button_pressed(self, event: Button.Pressed) ->
None:
        self.exit(event.button.id)
if __name__ == "__main__":
    app = HelloWorld()
    app.run()
```

Con esta parte lo que hacemos es que aparezca un layout con un botón para cerrar y un mensaje de texto que dice hello world.

```
Screen {
    layout: grid;
    grid-size: 2;
    grid-gutter: 2;
    padding: 2;
}
#hello {
    width: 100%;
    height: 100%;
    column-span: 2;
    content-align: center bottom;
    text-style: bold;
}
Button {
    width: 100%;
```

```
    column-span: 2;
}
```

Esta parte de aquí es el css que hace que el texto y los botones estén en el centro de la pantalla y que se ajusten a la pantalla.

TEXTUAL CON SCRIPT

Ahora mostraré como he implementado textual con el script del juego. Debido a la naturaleza de textual y a mi falta de experiencia con esta herramienta me ha sido imposible hacer el juego entero. Únicamente he conseguido hacer que muestra las reglas y una carta.