MANUAL TECNICO

Tabla de contenido

Variables

```
# variables
negras = ['♠', '♣']  # cartas negras
rojas = ['♥', '♦']  # cartas rojas

baraja = []  # almacena las cartas pero no se ven en el menu
carta = ["1♥", "1♦", "1♠", "1♣", "2♥", "2♦", "2♠", "2♣", "3♥", "3♦", "3♠", "3♣", "4♥", "4♦", "4♠", "4♣", "5
         "5♠", "5♣", "6♥", "6♦", "6♠", "6♣", "7♥", "7♦", "7♠", "7♣", "8♥", "8♦", "8♠", "8♣", "9♥", "9♦", "9
         "10♥", "10♦", "10♠", "10♣", "J♥", "J♦", "J♠", "J♣", "Q♥", "Q♦", "Q♠", "Q♣", "K♥", "K♦", "K♠", "K♣"
pila1 = []
pila2 = []
pila3 = []
pila4 = []
pila5 = []
pila6 = []
pila7 = []
pila8 = []  # trebol
pila9 = []  # diamante
pila10 = []  # corazon
pila11 = []  # pica
pila12 = []  # cartas que tomamos de la baraja

ganadas = 0  # estadisticas ganadas
perdidas = 0  # estadisticas perdidas
```

métodos

```
# crear baraja
def crearBaraja():...


# repartir las cartas en cada mano
def repartir():...


# valida orden decendente de las cartas y signo para que intercale entre rojo y negro una a una
def validarMovimiento(carta_origen,
                      carta_destino=False):...


# valida orden decendente de las cartas y signo para que intercale entre rojo y negro al mover varias
def validarMovimientoCartas(cartas, destino=False):...


# mover cartas de las pilas
def moverPilas():...


# recorrer baraja y que cuando se acabe las cartas se reinicie la baraja
def recorrerBaraja():...


# mover varias cartas
def moverCartas():...


# Reiniciar juego
def jugarDeNuevo():...


# menu
def mostrarMenu(intentos):...
```

Crear Baraja

Es un método para crear la baraja con un ciclo “for’’ para meter 52 cartas de la variable “carta’’ en la pila “baraja’’ y que queden acomodadas de manera aleatoria.

```
def crearBaraja():
    for i in range(52):
        baraja.append(carta[i])
    random.shuffle(baraja)
```

Repartir

Este método va a repartir en las “manos” o mazos en los que en el juego de solitario al empezar son acomodados de 1 carta hasta 7 pero únicamente se puede tomar la última carta.

Mediante ciclos “for’’ repartimos cartas desde la pila “baraja” el mazo 1(1 carta), mazo 2(2 cartas) y así sucesivamente hasta 7.

```
def repartir():
    for i in range(1):  # mano1
        pila1.append(baraja.pop())
    for i in range(2):  # mano2
        pila2.append(baraja.pop())
    for i in range(3):  # mano3
        pila3.append(baraja.pop())
    for i in range(4):  # mano4
        pila4.append(baraja.pop())
    for i in range(5):  # mano5
        pila5.append(baraja.pop())
    for i in range(6):  # mano6
        pila6.append(baraja.pop())
    for i in range(7):  # mano7
        pila7.append(baraja.pop())
```

Validar Movimiento

Este método valida de forma descendente las cartas para los mazos del 1-7 y también verifica su color para que intercale entre rojo y negro. (Al mover 1 sola carta)

```
def validarMovimiento(carta_origen,
                      carta_destino=False):  # = False signinifica que

    if carta_destino == False:
        return True

    origenP = carta_origen[-1]  # palo de la carta origen
    origenN = carta_origen[0]  # numero de la carta origen

    destinoP = carta_destino[-1]  # palo de la carta destino
    destinoN = carta_destino[0]  # numero de la carta destino

    if origenN == 'K':
        origenN = 13
    if origenN == 'Q':
        origenN = 12
    if origenN == 'J':
        origenN = 11
    if destinoN == 'K':
        destinoN = 13
    if destinoN == 'Q':
        destinoN = 12
    if destinoN == 'J':
        destinoN = 11

    origenN = int(origenN)
    destinoN = int(destinoN)

    if origenP in rojas:
        if destinoP in negras:

            if destinoN == origenN + 1:
                return True
            else:
                return False

    if origenP in negras:
        if destinoP in rojas:
            if destinoN == origenN + 1:
                return True
            else:
                return False

    else:
        return False
```

Validar Movimiento Varias

Este método valida de forma descendente las cartas para los mazos del 1-7 y también verifica su color para que intercale entre rojo y negro. (Al mover varias cartas)

Este método utiliza el anterior método también “Validar movimiento”.

```
def validarMovimientoCartas(cartas, destino=False):
    if destino == False:
        return True

    if validarMovimiento(cartas[-1], cartas[-2]):
        return True
    return False
```

Mover Pilas

Mueve cartas de pila a pila (1 a 1) Validando primero la pila origen(cual carta deseamos mover) a pila de destino(al mazo donde se desea poner la carta)

La opción 0 es la opción de la pila “Baraja o pila 12” siendo esta pila de origen una pila que puede ser usada de origen, pero no de destino. En cada uno de los destinos valida si es posible realizar la acción.

De la pila de origen 1 a la 7 se pueden mover de igual manera siendo mutuamente orígenes y destinos, en todas utilizamos un if y else para validar si se puede realizar el movimiento en caso contrario mostrara un print indicando que no se puede realizar.

La pila 8,9,10,11 se usan para ordenar de forma ascendente las cartas del mismo signo, en estos if y else se va a validar si la pila esta vacía debe empezar específicamente por As(1) y su respectivo signo(8 trébol ,9 Diamante ,10 corazón ,11 Picas) en forma ascendente únicamente respetando el orden incluido el de J,Q, y K que en cada pila de origen se especifica su valor J=11, Q=12 y K=13.

Ejemplo de pila origen 0(Baraja o pila 12) a Destino 1(Mazo1)

```
if pilaOrigen == 0:  # baraja

    if len(pila12) == 0:
        print("No hay cartas en la baraja")
        return moverPilas()
    else:
        if pilaDestino == 1:
            if validarMovimiento(pila12[-1], pila1[-1] if len(pila1) > 0 else False):
                pila1.append(pila12.pop())
            else:
                print("No se puede mover esa carta ahi")
                return moverPilas()
```

Ejemplo cuando el destino es la pila 8(es igual para la 9, 10 y 11) donde se ordenan ascendentemente

```
elif pilaDestino == 8:
    if pila1[-1][-1] == "♠":
        if len(pila8) == 0:
            if pila1[-1] == "1♠":
                pila8.append(pila1.pop())
            else:
                print("No puedes poner una carta de diferente 1♠ de primera carta")
        else:
            value0 = pila1[-1][0]
            value1 = pila8[-1][0]
            if value0 == 'K':
                value0 = 13
            elif value0 == 'Q':
                value0 = 12
            elif value0 == 'J':
                value0 = 11
            else:
                value0 = int(value0)
            if value1 == 'K':
                value1 = 13
            elif value1 == 'Q':
                value1 = 12
            elif value1 == 'J':
                value1 = 11
            else:
                value1 = int(value1)

            if value1 == value0 - 1:
                pila8.append(pila1.pop())
            else:
                print("Carta no es la siguiente")
    else:
        print("no puedes poner esta carta")
```

Recorrer Baraja

Este método permite que después de repartir a la pila “baraja” se pueda mover a la pila 12 carta por carta para poder ser usada en el juego.

Si la baraja se queda sin cartas se reinicia la baraja

```
def recorrerBaraja():
    global baraja
    global pila12
    if len(baraja) == 0:
        for i in range(len(pila12)):
            baraja.append(pila12.pop())
    else:
        pila12.append(baraja.pop())
```

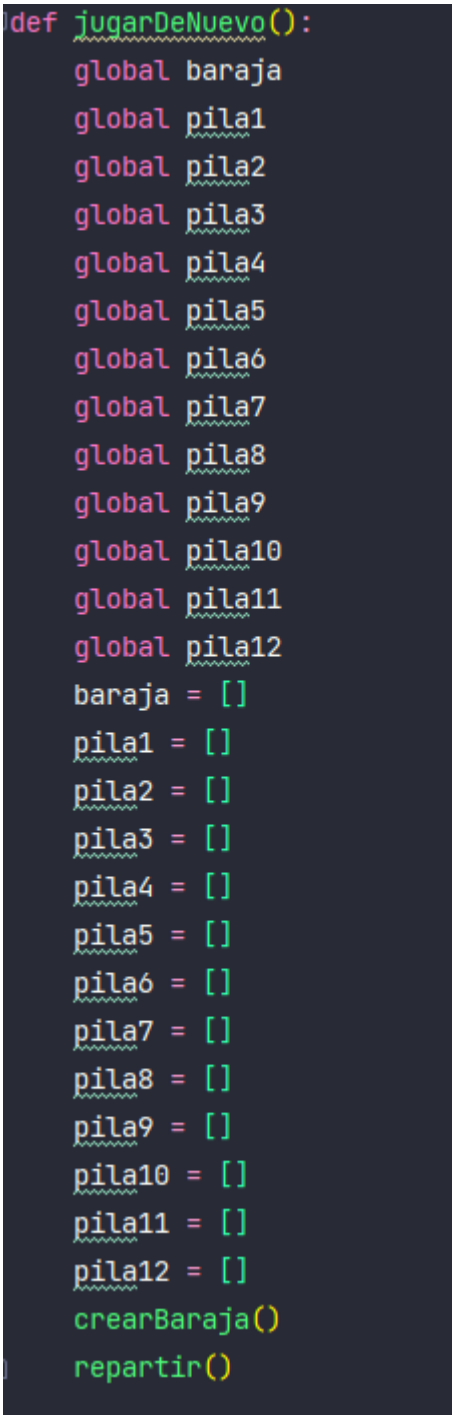
Mover cartas

Este método es igual al método Mover pilas, pero la única diferencia es que este método es para cuando se mueven varias cartas a la vez por ejemplo si tengo en el mazo1: 9,8,7 poder mover esas 3 cartas a otro mazo que tenga un 10. También para cuando se utiliza este método se pueda seleccionar cuantas cartas se puedan mover.

```
try:
    pilaOrigen = int(input("Pila origen: "))
    pilaDestino = int(input("Pila destino: "))
    nCartas = int(input("Numero de cartas: "))
except ValueError:
    input("Error, preciona para volver al menu")
    return mostrarMenu(intentos)
if pilaOrigen == 1:
    if len(pila1) == 0:
        print("No hay cartas para mover")
        return moverCartas()
    else:
        if pilaDestino == 2:
            for i in range(nCartas):
                if validarMovimiento(pila1[-1], pila2[-1] if len(pila2) > 0 else False):
                    pila2.append(pila1.pop())
                else:
                    print("No puedes poner esta carta")
                    continue
        elif pilaDestino == 3:
            for i in range(nCartas):
                if validarMovimiento(pila1[-1], pila3[-1] if len(pila3) > 0 else False):
                    pila3.append(pila1.pop())
                else:
                    print("No puedes poner esta carta")
                    continue
```

Jugar De Nuevo

Este método básicamente lo que hace es vaciar todas las pilas y utilizar de nuevo el método Crear Baraja y Repartir. En otras palabras, reinicia el juego.

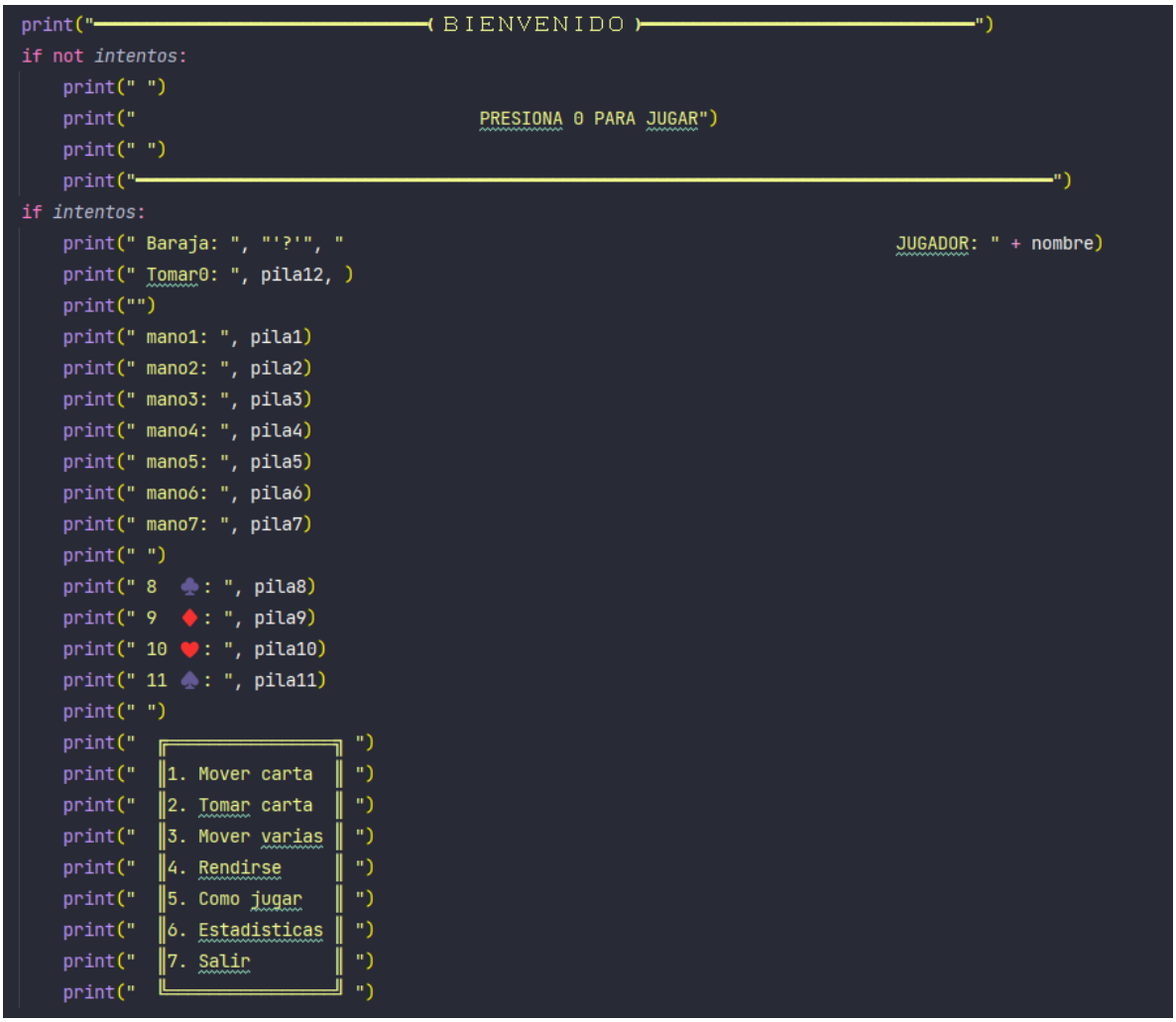


```
def jugarDeNuevo():
    global baraja
    global pila1
    global pila2
    global pila3
    global pila4
    global pila5
    global pila6
    global pila7
    global pila8
    global pila9
    global pila10
    global pila11
    global pila12
    baraja = []
    pila1 = []
    pila2 = []
    pila3 = []
    pila4 = []
    pila5 = []
    pila6 = []
    pila7 = []
    pila8 = []
    pila9 = []
    pila10 = []
    pila11 = []
    pila12 = []
    crearBaraja()
    repartir()
```

menú

Imprime todos los datos necesarios para el usuario pueda jugar y entender el juego

También muestra las opciones en las cuales se pueden Utilizar mediante números:



¡¡¡Mostrar si gana!!!

En un if si se cumple que las pilas 8,9,10 y 11 están llenas (13 cartas cada una) entonces gana la partida.

Y guarda los datos en un .txt con la opción “file.” Con nombre y su partida ganada y sus partidas perdidas.

Luego le pregunta si desea jugar de nuevo y en caso de que la opción sea S aplica el método Jugar De Nuevo y si no cierra el programa

También valida que si las pilas 8,9,10 y 11 tienen 13 cartas no pueden recibir mas cartas

```
if len([pila8, pila9, pila10, pila11]) == 13:
    ganadas += 1
    print("Ganaste")
    file = open("puntaje.txt", "w")
    file.write("Ganaste")
    file.write("Total ganadas: " + str(ganadas))
    file.write("Total perdidas: " + str(perdidas))
    file.close()
    print("Desea jugar de nuevo? (s/n)")
    respuesta = input()
    if respuesta == "s":
        jugarDeNuevo()
    else:
        exit()
elif len(pila8) >= 14:
    print("No puedes poner mas de 13 cartas en una pila♣")
elif len(pila9) >= 14:
    print("No puedes poner mas de 13 cartas en una pila♦")
elif len(pila10) >= 14:
    print("No puedes poner mas de 13 cartas en una pila♥")
elif len(pila11) >= 14:
    print("No puedes poner mas de 13 cartas en una pila♠")
```

Programa Principal

Aquí creamos una variable intentos para que cuando con la opción 0 demos jugar en el meno no muestre la opción más.

Luego en un ciclo While ponemos las distintas opciones del menú:

0: Para jugar y pedir el nombre y guardándolo en la variable nombre.

1: llama al método mover pilas para mover carta por carta.

2: llama al método recorrer baraja para tomar una carta de la baraja y mostrarla en el juego.

3: llama el método mover cartas para mover varias cartas a la vez.

4: Para reiniciar el juego y a la vez también en él .txt guarda el intento como una derrota por rendirse. Pregunta si desea jugar de nuevo o salirse.

5: Muestras las reglas del juego con un print.

6: Muestra las estadísticas del jugador llamando el .txt con el print(file.read())

7: Break para cerrar el programa

```
crearBaraja()
intentos = False
while True:
    mostrarMenu(intentos)
    print(" ")
    try:
        opcion = int(input(" opcion: "))
    except ValueError:
        print("Ingrese un numero")
        continue
    if opcion == 0:
        if intentos:
            print(" ")
        else:
            intentos = True
            print("Dijite su nombre: ")
            nombre = input()
            repartir()
    elif opcion == 1:
        moverPilas()
    elif opcion == 2:
        recorrerBaraja()
    elif opcion == 3:
        moverCartas()
```

```
elif opcion == 4:
    print("Perdiste")
    perdidas += 1
    file = open("puntaje.txt",
                "a")  # file = open("nombre doc", "w para sobreescribir y a para agregar permanente")
    file.write(nombre)
    file.write(": Total ganadas: " + str(ganadas))
    file.write(" Total perdidas: " + str(perdidas))
    file.write("\n")  # salto de linea
    file.close()
    print("Desea jugar de nuevo? (s/n)")
    respuesta = input()
    if respuesta == "s":
        jugarDeNuevo()
    else:
        exit()
    jugarDeNuevo()
elif opcion == 5:
    print(" ")
    print(
        "  ╔═══════════════════════════════════════════════════════════════════════════════════════════════════════════════════════╗ ")
    print(
        "  ║1. El objetivo del juego es colocar todas las cartas en las pilas 8,9,10,11 en su orden acendente    1=As          ║ ")
    print(
        "  ║        ejemplo:  8♠: 1♠,2♠,3♠,4♠,5♠,6♠,7♠,8♠,9♠,10♠,J♠,Q♠,K♠                                                          ║ ")
    print(
        "  ║2. Para mover una carta, debes seleccionar la opcion 1 y escoger la pila de origen y la pila de destino.              ║ ")
    print(
        "  ║        pila de origen: donde esta la carta que quieres mover                                                         ║ ")
    print(
        "  ║        pila de destino: a donde quieres que este la carta que estas moviendo                                        ║ ")
    print(
        "  ║3. Para mover varias cartas, debes seleccionar la opcion 3                                                            ║ ")
    print(
        "  ║4. Para tomar una carta, debes seleccionar la opcion 2 y la carta tomara una carta de la varaja, puedes hacer         ║ ")
    print(
        "  ║  esto varias veces para recorrer la baraja                                                                           ║ ")
    print(
        "  ║5. Para rendirse, debes seleccionar la opcion 4                                                                       ║ ")
    print(
        "  ║6. Para salir, debes seleccionar la opcion 6.                                                                         ║ ")
    print(
        "  ║                                                                                                                      ║ ")
    print(
        "  ╚══════════════════════════════════════════════════════════════════════════════════════════════════════════════════════╝ ")
    print(" ")
    print("Presione enter para volver al menu")
    input()
```

```
elif opcion == 6:
    file = open("puntaje.txt")
    print(file.read())
    file.close()
    print("Presione enter para volver al menu")
elif opcion == 7:
    break
else:
    print("Opcion incorrecta")
input("Pulse enter para continuar...")
```