

21 DE NOVIEMBRE DE 2025

JUEGO DE LAS DAMAS ESPAÑOLAS:
PYTHON
ENRIQUE GONZÁLEZ



CAMPUSFP
1º DAM GETAFE

Contenido

APP.PY.....	4
CODIGO:.....	4
DAMAS_CORE.PY.....	6
CODIGO:.....	6
COMPROBACIÓN:	13

ENRIQUE GONZÁLEZ
1º DAM GETAFE

Damas (con funciones y módulos, usando NumPy)

Objetivo

Implementar un juego de Damas españolas en consola:

Tablero 8×8 (solo se juega en casillas oscuras).

Dos jugadores: Jugador 1 y Jugador 2.

Piezas:

Peón J1 = 1, Rey J1 = 2

Peón J2 = 3, Rey J2 = 4

Vacía = 0

Peones: se mueven 1 diagonal hacia delante (J1 hacia arriba, J2 hacia abajo).

Reyes: se mueven 1 diagonal en ambas direcciones.

Capturas obligatorias (saltar en diagonal sobre pieza rival a un hueco).

Cadenas de captura: si tras capturar puede seguir capturando la misma pieza, debe hacerlo en el mismo turno.

Coronación: un peón se convierte en Rey al llegar a la fila opuesta.

Fin de partida: gana quien deja al rival sin movimientos o sin piezas.

Interfaz en consola, dos jugadores humanos.

Requisitos técnicos

Usar NumPy para el tablero.

ENRIQUE GONZÁLEZ
1º DAM GETAFE

Separar en módulos:

damas_core.py: toda la lógica (funciones puras y utilidades).

app.py: menú/bucle de partida e interacción por teclado.

Control de flujo básico (while, for, if), sin break/continue.

Estructura mínima
damas/
 damas_core.py
 app.py

APP.PY

CODIGO:

```
import numpy as np

import damas_core

#FUNCION PARA PINTAR EL TABLERO DE FORMA LEGIBLE

def pintar_tablero(tablero):

    print(" 0 1 2 3 4 5 6 7")

    print(" -----")

    for fila in range(8):

        linea = str(fila) + " | "

        for columna in range(8):

            valor = tablero[fila, columna]

            if valor == 0:

                simbolo = "."

            elif valor == 1:

                simbolo = "o"

            elif valor == 2:

                simbolo = "O"

            elif valor == 3:

                simbolo = "x"

            elif valor == 4:

                simbolo = "X"

            else:

                simbolo = "?"

            linea = linea + simbolo + " "

    print(linea)
```

ENRIQUE GONZÁLEZ
1º DAM GETAFE

```
print(linea)
print(" -----\\n")

#FUNCION DEL FUNCIONAMIENTO DEL PROGRAMA

def programaFinal():

    tablero = np.zeros((8, 8))

    damas_core.colocar_fichas(tablero)

    pintar_tablero(tablero)

    terminado = False

    while not terminado:

        print("Turno Jugador 1")

        damas_core.jugador1(tablero)

        pintar_tablero(tablero)

        if damas_core.verSiHaTerminado(tablero):

            terminado = True

        else:

            print("Turno Jugador 2")

            damas_core.jugador2(tablero)

            pintar_tablero(tablero)

            if damas_core.verSiHaTerminado(tablero):

                terminado = True

#LLAMAR A LA FUNCION PARA EJECUTAR EL PROGRAMA

programaFinal()
```

DAMAS_CORE.PY

CODIGO:

```
import numpy as np

#CONSTANTE PARA LIMITAR MOVIMIENTOS DENTRO DEL TABLERO
MAXTABLERO = 7

#FUNCION QUE REVISA SI UNA POSICION ESTA DENTRO DEL TABLERO
def dentro(f, c):
    return f >= 0 and f <= MAXTABLERO and c >= 0 and c <= MAXTABLERO

#FUNCION PARA REALIZAR MOVIMIENTO Y COMIDAS ENCENDIDAS
def mover_y_comer(tablero, fila, columna, df, dc, enemigos, nuevo_valor):
    #CALCULA LA PRIMERA CASILLA A LA QUE SE QUIERE MOVER
    fila_nueva = fila + df
    columna_nueva = columna + dc

    #SI ESTA FUERA DEL TABLERO NO SE MUEVE
    if not dentro(fila_nueva, columna_nueva):
        print("ESTA FICHA NO SE PUEDE MOVER HACIA ESA DIRECCION")
        return

    #ELIMINA LA FICHA ORIGINAL
    tablero[fila, columna] = 0

    #MIENTRAS ENCUENTRE ENEMIGOS SIGUE AVANZANDO
    seguir = True
    while seguir:
        if dentro(fila_nueva, columna_nueva):
            if tablero[fila_nueva, columna_nueva] in enemigos:
                print("HAS COMIDO UNA FICHA DEL OPONENTE")
                tablero[fila_nueva, columna_nueva] = 0
            fila_nueva = fila_nueva + df
```

ENRIQUE GONZÁLEZ
1º DAM GETAFE

```
columna_nueva = columna_nueva + dc
else:
    seguir = False
else:
    seguir = False

#AL FINAL COLOCA LA FICHA SOLO SI LA CASILLA ES VALIDA

if dentro(fila_nueva, columna_nueva):
    tablero[fila_nueva, columna_nueva] = nuevo_valor
else:
    print("EL MOVIMIENTO TERMINO FUERA DEL TABLERO")

#FUNCION PARA VER SI UNA FICHA DEBE CORONARSE

def coronar(jugador, fila):
    if jugador == 1 and fila == 0:
        return 2
    if jugador == 2 and fila == 7:
        return 4
    return None

#FUNCION QUE COLOCA LAS FICHAS EN EL TABLERO CON UN PATRON INICIAL

def colocar_fichas(tablero):
    fila1 = 0
    fila2 = 1
    fila3 = 2
    fila4 = 5
    fila5 = 6
    fila6 = 7

    patron = np.tile([3, 0], 4) #PATRON PARA PEONES DEL JUGADOR 2
    tablero[fila2] = patron
```

ENRIQUE GONZÁLEZ
1º DAM GETAFE

```
patron = np.tile([1, 0], 4) #PATRON PARA PEONES DEL JUGADOR 1
tablero[fila4] = patron
tablero[fila6] = patron
```

```
patron2 = np.tile([0, 1], 4) #PATRON PARA EL JUGADOR 1
tablero[fila5] = patron2
```

```
patron2 = np.tile([0, 3], 4) #PATRON PARA EL JUGADOR 2
tablero[fila1] = patron2
tablero[fila3] = patron2
```

```
print(tablero)
```

#FUNCION DEL JUGADOR 1: MANEJA PEONES, REYES, MOVIMIENTOS Y CORONACION

```
def jugador1(tablero):
    encontrado = False
    while not encontrado:
        fila = int(input("Fila: (de 0 a 7) "))
        while fila < 0 or fila > 7:
            print("ESTA FILA NO ES VALIDA")
        fila = int(input("Fila: (de 0 a 7) "))
        columna = int(input("Columna: (de 0 a 7) "))
        while columna < 0 or columna > 7:
            print("COLUMNNA NO VALIDA")
        columna = int(input("Columna: (de 0 a 7) "))
        pieza = tablero[fila, columna]
        #PEON SIMPLE DEL JUGADOR 1
        if pieza == 1:
```

```
encontrado = True

lado = input("HAY UN PEON, A QUE LADO QUIERES MOVER? ('i'=izquierda  
'd'=derecha) ")

enemigos = [3, 4]

df = -1

if lado == "i":

    dc = -1

elif lado == "d":

    dc = 1

else:

    print("DIRECCION NO VALIDA")

return

nuevo_valor = 1

mover_y_comer(tablero, fila, columna, df, dc, enemigos, nuevo_valor)

#REvisa CORONACION

for i in range(8):

    for j in range(8):

        if tablero[i, j] == 1:

            corona = coronar(1, i)

            if corona is not None:

                tablero[i, j] = corona

#REY DEL JUGADOR 1

elif pieza == 2:

    encontrado = True

    lado = input("HAY UN REY, A QUE LADO QUIERES MOVER? ('ai','ad','li','ld') ")

    enemigos = [3, 4]

    df = 0

    dc = 0
```

ENRIQUE GONZÁLEZ
1º DAM GETAFE

```
if lado == "ai":  
    df = -1; dc = -1  
  
elif lado == "ad":  
    df = -1; dc = 1  
  
elif lado == "li":  
    df = 1; dc = -1  
  
elif lado == "ld":  
    df = 1; dc = 1  
  
else:  
    print("DIRECCION NO VALIDA")  
    return  
  
mover_y_comer(tablero, fila, columna, df, dc, enemigos, 2)  
  
else:  
    print("NO HAY PEON NI REY EN ESA POSICION")  
  
#FUNCION DEL JUGADOR 2: MANEJA PEONES, REYES, MOVIMIENTOS Y  
CORONACION  
  
def jugador2(tablero):  
    encontrado = False  
  
    while not encontrado:  
        fila = int(input("Fila: (de 0 a 7) "))  
  
        while fila < 0 or fila > 7:  
            print("ESTA FILA NO ES VALIDA")  
            fila = int(input("Fila: (de 0 a 7) "))  
  
        columna = int(input("Columna: (de 0 a 7) "))  
  
        while columna < 0 or columna > 7:  
            print("COLUMNA NO VALIDA")  
            columna = int(input("Columna: (de 0 a 7) "))
```

```
pieza = tablero[fila, columna]

#PEON DEL JUGADOR 2

if pieza == 3:

    encontrado = True

    lado = input("HAY UN PEON, A QUE LADO QUIERES MOVER? ('i'=izquierda
'd'=derecha) ")

    enemigos = [1, 2]

    df = 1

    if lado == "i":

        dc = -1

    elif lado == "d":

        dc = 1

    else:

        print("DIRECCION NO VALIDA")

    return

nuevo_valor = 3

mover_y_comer(tablero, fila, columna, df, dc, enemigos, nuevo_valor)

#CORONACION

for i in range(8):

    for j in range(8):

        if tablero[i, j] == 3:

            corona = coronar(2, i)

            if corona is not None:

                tablero[i, j] = corona

#REY DEL JUGADOR 2

elif pieza == 4:

    encontrado = True

    lado = input("HAY UN REY, A QUE LADO QUIERES MOVER? ('ai','ad','li','ld') ")
```

```
enemigos = [1, 2]

df = 0; dc = 0

if lado == "ai":

    df = -1; dc = -1

elif lado == "ad":

    df = -1; dc = 1

elif lado == "li":

    df = 1; dc = -1

elif lado == "ld":

    df = 1; dc = 1

else:

    print("DIRECCION NO VALIDA")

    return

mover_y_comer(tablero, fila, columna, df, dc, enemigos, 4)

else:

    print("NO HAY PEON NI REY EN ESA POSICION")

#FUNCION QUE MIRA SI YA NO QUEDAN PIEZAS DE ALGUN JUGADOR

def verSiHaTerminado(tablero):

    if not np.any((tablero == 1) | (tablero == 2)):

        print("GANADOR: JUGADOR 2")

        return True

    if not np.any((tablero == 3) | (tablero == 4)):

        print("GANADOR: JUGADOR 1")

        return True

    return False
```

COMPROBACIÓN:

Tablero con numpy y representación gráfica con función pintar_tablero:

```
PS C:\Users\CampusFP\Desktop\Juego de
[[0. 3. 0. 3. 0. 3. 0. 3.]
 [3. 0. 3. 0. 3. 0. 3. 0.]
 [0. 3. 0. 3. 0. 3. 0. 3.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 1. 0. 1. 0. 1. 0.]
 [0. 1. 0. 1. 0. 1. 0. 1.]
 [1. 0. 1. 0. 1. 0. 1. 0.]]
 0 1 2 3 4 5 6 7
-----
0 | . x . x . x . x
1 | x . x . x . x .
2 | . x . x . x . x
3 | . . . . . . .
4 | . . . . . . .
5 | o . o . o . o .
6 | . o . o . o . o
7 | o . o . o . o .
-----
```

Movimientos ambos jugadores

Turno Jugador 1

Fila: (de 0 a 7) 5

Columna: (de 0 a 7) 0

HAY UN PEON, A QUE LADO QUIERES MOVER? ('i'=izquierda 'd'=derecha) d

0 1 2 3 4 5 6 7

0		.	x	.	x	.	x	.	x
1		x	.	x	.	x	.		
2		.	x	.	x	.	x	.	
3		
4		.	o	
5		.	.	o	.	o	.	o	.
6		.	o	.	o	.	o	.	o
7		o	.	o	.	o	.	o	.

Turno Jugador 2

Fila: (de 0 a 7) 2

Columna: (de 0 a 7) 1

HAY UN PEON, A QUE LADO QUIERES MOVER? ('i'=izquierda 'd'=derecha) i

0 1 2 3 4 5 6 7

0		.	x	.	x	.	x	.	x
1		x	.	x	.	x	.	x	.
2		.	.	x	.	x	.	x	.
3		x	
4		.	o	
5		.	.	o	.	o	.	o	.
6		.	o	.	o	.	o	.	o
7		o	.	o	.	o	.	o	.

Turno Jugador 1

ENRIQUE GONZÁLEZ
1º DAM GETAFE

Creación de REY

```
Turno Jugador 2
Fila: (de 0 a 7) 3
Columna: (de 0 a 7) 0
HAY UN PEON, A QUE LADO QUIERES MOVER? ('i'=izquierda 'd'=derecha) d
 0 1 2 3 4 5 6 7
-----
0 | . 0 . x . x . x
1 | x . . . x . x .
2 | . x . . . . . o
3 | . . . . . x .
4 | . x . o . . . .
5 | o . . . . o . .
6 | . . . o . o . o
7 | o . o . o . o .
```

Movimiento Rey

(ai y ad = movimientos para arriba) (li y ld = movimientos para abajo)

```
0 1 2 3 4 5 6 7
-----
0 | . 0 . . . x . x
1 | . . o . x . . .
2 | . . . . . . . o
3 | . . o . o . x .
4 | . . . . . . .
5 | . . . . . . .
6 | . . . . . o . o
7 | o . o . o . o .
-----
Turno Jugador 1
Fila: (de 0 a 7) 1
Columna: (de 0 a 7) 0
NO HAY PEON NI REY EN ESA POSICION
Fila: (de 0 a 7) 0
Columna: (de 0 a 7) 1
HAY UN REY, A QUE LADO QUIERES MOVER? ('ai','ad','li','ld') li
 0 1 2 3 4 5 6 7
-----
0 | . . . . x . x
1 | 0 . o . x . . .
2 | . . . . . . . o
3 | . . o . o . x .
4 | . . . . . . .
5 | . . . . . . .
6 | . . . . . o . o
7 | o . o . o . o .
```

ENRIQUE GONZÁLEZ
1º DAM GETAFE