Caracas, 07 de enero de 2025
Algoritmos y Programación
Profesor: Jose Pacheco

# 2048

Link github:
https://github.com/alehernandezbucab/2048_game.git

Integrantes:
Alejandra Hernández  CI: 30751083
Paola Marturet CI: 31229493
Oliver Guillen CI: 31047247

El juego 2048 es un popular juego de lógica y habilidades matemáticas. El objetivo es combinar fichas numéricas en una cuadrícula de 4x4 hasta alcanzar la ficha con el valor 2048. Los jugadores deben deslizar las fichas en diferentes direcciones (arriba (w), abajo(s), izquierda(a) y derecha(d)) para combinarlas y sumar sus valores.

En este proyecto, presentamos una implementación del juego 2048 utilizando el lenguaje de programación Python. El código abarca diversas funcionalidades esenciales para el funcionamiento del juego, desde la inicialización y gestión de la matriz de juego, hasta el manejo de movimientos de las fichas y la verificación de condiciones de victoria o derrota. Además, se incluye un sistema para almacenar y leer los resultados de las partidas, permitiendo a los jugadores mantener un registro de su desempeño.

## Código:

```
import random
import os
import copy
import datetime
```

Random:  este módulo se usa para generar números aleatorios.
Os:  este módulo se usa en este código para la limpieza de la pantalla del terminal.
Copy: este módulo se usa en el código para crear una copia profunda del tablero del juego, es útil para evitar modificaciones
Datetime: se utiliza para trabajar con fechas y horas, se usa para registrar la fecha y hora en que se guardan los resultados del juego.

```
def crear_matriz():
    game = []
    for i in range(0,4):
        game.append([0]*4)
    return game

def mostrar_matriz(game):
    tablero = copy.deepcopy(game)
    for i in range(len(tablero)):
        for j in range(len(tablero[i])):
            if tablero[i][j] == 0:
                tablero[i][j] = " "

    print("-" * ((7 * len(tablero) + 1)))
    for i in range(len(tablero)):
        print("|", end="")
        for j in range(len(tablero[i])):
            print(f" {tablero[i][j]:^4} |", end="")
        print("\n" + "-" * ((7 * len(tablero) + 1)))
```

Crear matriz 4x4 inicializada con ceros.
Muestra la matriz del juego en un formato legible

UCAB Universidad Católica ANDRÉS BELLO
Dirección de Identidad y Misión

```python
def espacios_vacios(game):
    vacios = []

    for i in range(len(game)):
        for j in range(len(game[i])):
            if game[i][j] == 0:
                vacios.append((i, j))
    return vacios

def rellenar_espacio_vacio(game, vacios):
    n = random.choice([2,2,2,2,2,2])

    casillas = random.choice(vacios)
    game[casillas[0]][casillas[1]] = n
    return game
```

Encuentra posiciones vacías en la matriz

Rellena una posición vacía al azar con un número 2

```python
def mover_arriba(game):
    for i in range(4):
        for j in range(4):
            if game[i][j] == 0:
                for k in range(i+1, 4):
                    if game[k][j] != 0:
                        game[i][j] = game[k][j]
                        game[k][j] = 0
                        break
    for i in range(3):
        for j in range(4):
            if game[i][j] == game[i+1][j]:
                game[i][j] *= 2
                game[i+1][j] = 0
    for i in range(4):
        for j in range(4):
            if game[i][j] == 0:
                for k in range(i+1, 4):
                    if game[k][j] != 0:
                        game[i][j] = game[k][j]
                        game[k][j] = 0
                        break
    return game
```

```python
def mover_abajo(game):
    for i in range(3, -1, -1):
        for j in range(4):
            if game[i][j] == 0:
                for k in range(i-1, -1, -1):
                    if game[k][j] != 0:
                        game[i][j] = game[k][j]
                        game[k][j] = 0
                        break
    for i in range(3, 0, -1):
        for j in range(4):
            if game[i][j] == game[i-1][j]:
                game[i][j] *= 2
                game[i-1][j] = 0
    for i in range(3, -1, -1):
        for j in range(4):
            if game[i][j] == 0:
                for k in range(i-1, -1, -1):
                    if game[k][j] != 0:
                        game[i][j] = game[k][j]
                        game[k][j] = 0
                        break
    return game
```

```python
def mover_izquierda(game):
    for i in range(4):
        for j in range(4):
            if game[i][j] == 0:
                for k in range(j+1, 4):
                    if game[i][k] != 0:
                        game[i][j] = game[i][k]
                        game[i][k] = 0
                        break
    for i in range(4):
        for j in range(3):
            if game[i][j] == game[i][j+1]:
                game[i][j] *= 2
                game[i][j+1] = 0
    for i in range(4):
        for j in range(4):
            if game[i][j] == 0:
                for k in range(j+1, 4):
                    if game[i][k] != 0:
                        game[i][j] = game[i][k]
                        game[i][k] = 0
                        break
    return game
```

```python
def mover_derecha(game):
    for i in range(4):
        for j in range(3, -1, -1):
            if game[i][j] == 0:
                for k in range(j-1, -1, -1):
                    if game[i][k] != 0:
                        game[i][j] = game[i][k]
                        game[i][k] = 0
                        break
    for i in range(4):
        for j in range(3, 0, -1):
            if game[i][j] == game[i][j-1]:
                game[i][j] *= 2
                game[i][j-1] = 0
    for i in range(4):
        for j in range(3, -1, -1):
            if game[i][j] == 0:
                for k in range(j-1, -1, -1):
                    if game[i][k] != 0:
                        game[i][j] = game[i][k]
                        game[i][k] = 0
                        break
    return game
```

Movimientos de las celdas en cuatro direcciones, si hay el mismo valor adyacente, se combinan

```python
def ganador(game):
    for i in range(4):
        for j in range(4):
            if game[i][j] == 2048:
                return True
    return False

def sin_movimientos(game):
    for i in range(4):
        for j in range(4):
            if game[i][j] == 0:
                return False
    for i in range(3):
        for j in range(3):
            if game[i][j] == game[i+1][j] or game[i][j] == game[i][j+1]:
                return False
    for i in range(3):
        if game[i][3] == game[i+1][3]:
            return False
    for j in range(3):
        if game[3][j] == game[3][j+1]:
            return False
    return True
```

Verifica si alguna celda tiene el valor de 2048

Verifica si no hay movimientos posibles

```python
def guardar_resultados(nombre_jugador, tablero, movimientos,punt):
    with open("resultados.txt", "a") as f:
        f.write(f"Nombre: {nombre_jugador}\n")
        f.write(f"Fecha: {datetime.datetime.now()}\n")
        f.write("Estado final del tablero:\n")
        for fila in tablero:
            f.write("\t".join(str(num) for num in fila) + "\n")
        f.write(f"Movimientos: {movimientos}, Puntos:{punt}\n\n")

def leer_resultados():
    try:
        with open("resultados.txt", "r") as f:
            print(f.read())
    except FileNotFoundError:
        print("No hay resultados guardados.")

def sumar_matriz(game):
    suma = 0
    for i in range(4):
        for j in range(4):
            suma += game[i][j]
    return suma
```

Guarda resultados en un archivo

Lee esos resultados

Suma los valores de la matriz

```python
from game import *
```

Importamos funciones del archivo game donde se encuentra el juego 2048

```
op = 0
while(op != 3):
    menu()
    op = int(input("Ingrese una opcion: "))
```

Bucle mientras la opción no sea 3 que es salir.
Se pide al usuario colocar una opción.

```
if(op == 1):
    nombre = nombre_jugador()
    game = crear_matriz()
    vacios = espacios_vacios(game)
    game = rellenar_espacio_vacio(game, vacios)
    movimientos = 0
    num = 1
    salir = True
```

Si el jugador elige opción 1 se inicia nueva partida del juego
Se solicita al jugador y se inicializan game, vacios, movimientos, num, salir

```
while salir:
    os.system("cls")
    bienvenido(nombre)
    if num != 0:
        vacios = espacios_vacios(game)
        game = rellenar_espacio_vacio(game, vacios)
```

Mientras el jugador no se salga del juego, se sigue ejecutando y aquí viene la lógica del juego. Se actualiza la matriz con una nueva ficha en una posición vacía

```
jugada = ""
while jugada not in ["w", "a", "s", "d", "p"]:
    os.system("cls")
    bienvenido(nombre)
    mostrar_matriz(game)
    print(f"Movimientos: {movimientos}, Puntos: {sumar_matriz(game)}")
    jugada = input("Ingrese una jugada (w/a/s/d) o presione (p) para salir: ")

else:
    if jugada == "w" or jugada == "W":
        game = mover_arriba(game)
        movimientos += 1
    elif jugada == "a" or jugada == "A":
        game = mover_izquierda(game)
        movimientos += 1
    elif jugada == "s" or jugada == "S":
        game = mover_abajo(game)
        movimientos += 1
    elif jugada == "d" or jugada == "D":
        game = mover_derecha(game)
        movimientos += 1
    elif jugada == "p" or jugada == "P":
        salir = input("¿Seguro que quieres salir? (s/n): ")
        if salir.lower() == "s":
            print("Gracias por jugar!")
            salir = False
            guardar_resultados(nombre, game, movimientos,sumar_matriz(game))
            break
        else:
            continue
```

Se solicita un movimiento **arriba (w), abajo(s), izquierda(a), derecha(d) o salir (p)**, dependiendo del caso, se llama a la función correspondiente para mover las fichas en la dirección indicada o se procesa la opción de salir.

```
if ganador(game):
    os.system("cls")
    mostrar_matriz(game)
    print(f"Felicidades {nombre}, has ganado!")
    guardar_resultados(nombre, game, movimientos,sumar_matriz(game))
    break

if len(espacios_vacios(game)) == 0 and sin_movimientos(game) == True:
    os.system("cls")
    mostrar_matriz(game)
    print(f"{nombre}, Has perdido!")
    guardar_resultados(nombre, game, movimientos,sumar_matriz(game))
    break
```

Si el jugador alcanza una ficha de valor 2048, se muestra el tablero final, un mensaje de **felicitación**, y se guardan los resultados
Si no hay más movimientos posibles, se muestra el tablero final, un mensaje de **derrota**, y se guardan los resultados.

```
elif(op == 2):
    leer_resultados()
```

Si se elige opción 2, se leen y muestran los resultados

```
elif(op == 3):
    print("Salir, Gracias por Jugar :)")
    break
else:
    print("Opcion Invalida")
    continue
```

Si se elige opción 3, se muestra un mensaje de despedida y se sale del bucle principal
Si el  jugador ingresa una opción inválida, se muestra un mensaje de error y se vuelve a mostrar el menú