PUZZLE DESLIZANTE

PRÁCTICA 1

Yolanda Gómez Henche - Borja Martín Alonso | LP - Python | 30/04/2019

Manual del puzzle:

Nuestro puzzle funciona de la siguiente manera:

Para empezar, hemos creado un Python file, en el cual primero creamos una lista de números enteros con 16 números, creamos dos variables **FILAS y COLUMNAS** ambas con un valor de 4 para crear nuestra matriz que más tarde dará como resultado el puzzle.

A continuación, se crean todas las funciones necesarias para llevar a cabo una partida. Las cuales vamos a nombre una a una con una pequeña información sobre lo que hacen:

1. ***sonIguales*** en la cual lo que vamos a comprobar es que si el vector el cual rellenamos con un bucle for en rango 16 es igual a la lista dada.

2. ***buscarHueco*** dándole como argumento la Matriz lo que va a hacer es, recorriendo la matriz, mira toda las posiciones comprobando dónde está situado el hueco.

3. ***dameAdyacentes*** pasándoles como argumento la matriz, la fila del hueco y la columna del hueco: con una serie de if else anidados buscamos que nos retorne los valores adyacentes a la posición del hueco.

4. ***buscarAdyacente*** pasándole como argumento lo mismo que en la anterior, pero le pasamos adicionalmente el número adyacente escogido por el usuario para poder saber en qué posición se colocará el número adyacente.

Ahora pasamos a crear la clase Puzzle, en la cual, lo primero que vamos a meter es un constructor, sus métodos ***get*** pertinentes y un método ***toString***. Seguidamente, creamos otra serie de funciones importantes a la hora de realizar el algoritmo.

1. ***mostrarAdyacentes***: imprimirá por pantalla cuáles son los números adyacentes a la posición libre.

2. ***esAdyacente***: comprobará si el número escogido es o no adyacente al hueco.

3. ***esSolucion***: comprueba si la matriz modificada es igual a la matriz resultante.

4. ***recolocarPiezas***: reordenar las piezas que se van a mover de la matriz, adyacente escogido y hueco, para generar la matriz modificada al realizar dicha recolocación de números.

Una vez creado este Python file creamos una clase main en la cual vamos a probar todos estos métodos para poder jugar una partida.

Al principio damos una visión general del puzzle resuelto, explicamos en que consiste el juego para que la persona que vaya a realizar la partida entienda cuál es el fin de este juego y si desea resolver el puzzle mediante las teclas “s” , que representa el SÍ, o “n”, que representa el NO.

```
Esta es la vista del puzzle resuelto

[[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]
 [13 14 15  0]]


Usted debe generar ese mismo puzzle a partir de otro diferente cuyas posiciones
estan colocadas aleatoriamente. ¿Desea resolver el puzzle? [s/n]:
```

En el caso de que el usuario acepte el reto de intentar resolver el juego, éste escribe su nombre de usuario y, a continuación, aumentamos el número de partidas que realizará dicho usuario y aparecerá el puzzle desordenado con la posición del hueco y sus números antecedentes, el cual el usuario deberá elegir uno de ellos para poder hacer la recolocación para la nueva matriz de números generada con ese movimiento (aumentando así el número de movimientos realizados por el usuario).

```
Usted debe generar ese mismo puzzle a partir de otro diferente cuyas posiciones
estan colocadas aleatoriamente. ¿Desea resolver el puzzle? [s/n]: s

Nombre de Usuario: Yolanda

[[ 0  3 11  5]
 [ 1 15  4  2]
 [12 13  6 14]
 [ 9  8  7 10]]

Posicion libre: (0, 0)
Numeros adyacentes al hueco: 3 1
De esos numeros adyacentes, ¿cual desea escoger?: 3

[[ 3  0 11  5]
 [ 1 15  4  2]
 [12 13  6 14]
 [ 9  8  7 10]]

Numero de movimientos realizados: 1
Posicion libre: (0, 1)
Numeros adyacentes al hueco: 3 11 15
De esos numeros adyacentes, ¿cual desea escoger?: 11

[[ 3 11  0  5]
 [ 1 15  4  2]
 [12 13  6 14]
 [ 9  8  7 10]]

Numero de movimientos realizados: 2
Posicion libre: (0, 2)
Numeros adyacentes al hueco: 11 5 4
De esos numeros adyacentes, ¿cual desea escoger?:
```

Cuando el usuario lleva realizados 10 movimientos, o cualquier movimiento múltiplo de 10, el juego le pregunta si desea rendirse. Si el jugador afirma de que se rinde, el juego se acaba y aparecerá el siguiente mensaje: "¡Hasta la próxima!".

```
[[ 3 11  5  2]
 [ 9  1  0 14]
 [ 8 15  4  6]
 [12  7 13 10]]

Numero de movimientos realizados: 19
Posicion libre: (1, 2)
Numeros adyacentes al hueco: 5 1 14 4
De esos numeros adyacentes, ¿cual desea escoger?: 14


¿Desea rendirse?[s/n]: s

¡Hasta la proxima!
```

En caso de que no lo afirme, el juego continuará por el movimiento donde se había quedado pausado, para que el usuario siga intentándolo resolver.

```
[[ 3 11  5  2]
 [ 1 15  4 14]
 [ 9 12 13  6]
 [ 0  8  7 10]]

Numero de movimientos realizados: 9
Posicion libre: (3, 0)
Numeros adyacentes al hueco: 9 8
De esos numeros adyacentes, ¿cual desea escoger?: 8


¿Desea rendirse?[s/n]: n

[[ 3 11  5  2]
 [ 1 15  4 14]
 [ 9 12 13  6]
 [ 8  0  7 10]]

Numero de movimientos realizados: 10
Posicion libre: (3, 1)
Numeros adyacentes al hueco: 12 8 7
De esos numeros adyacentes, ¿cual desea escoger?: 12
```

Al realizar la nueva recolocación de las piezas del puzzle, el juego comprueba que la matriz modificada es exactamente igual que la resultante y, en caso de que sea cierta, el juego se acabará con el mensaje: "¡Hasta la próxima!".

MEJORAS

Una de las mejoras que hemos incluido en el juego del Puzzle Deslizante es utilizar un fichero denominado "ganadores.txt" donde guardamos aquellos usuarios que hayan resuelto el puzzle y el número de movimientos que han necesitado para resolverlo. Utilizamos una variable "partidas" que nos servirá para decir cuántas partidas ha realizado dicho usuario y, si lleva más de una partida jugada, sabremos que el jugador ha querido reintentar resolver el puzzle. En caso de conseguir de nuevo el puzzle resuelto, pero esta vez con menor número de movimientos que las anteriores partidas, se guardará esa nueva cifra de movimientos para dicho usuario ganador.

```
if matriz.essolucion():
    print('¡Genial has resuelto el puzzle en ',matriz.getMovimientos(),' movimientos!')
    if partidas == 1:
        archivo = open("ganadores.txt","w")
        archivo.writelines(usuario,' --> ',str((matriz.getMovimientos)))
        archivo.close()
    else:
        if matriz.getMovimientos < movimientosMejorPartida:
            movimientosMejorPartida = matriz.getMovimientos()
            archivo = open("ganadores.txt","a")
            archivo.writelines(usuario,' --> ',str((movimientosMejorPartida)))
            archivo.close()
    reintento = ', ¿quiere volver a reconstruir el puzzle?[s/n]: '
    letra1 = input(usuario,reintento)
    if letra1 == 'n' or 'N':
        break
```