

FP 2

Práctica 1

Fecha de entrega: 18 de abril de 2021

Puzzle con matrices.

1. Descripción de aspectos básicos del juego

Este juego consiste en un puzzle de ingenio para reconstruir una imagen (en adelante modelo objetivo) a partir de otra inicial. En función de las posibles acciones que se pueden realizar se dispone de dos modos de juego. En el modo 1D se pueden realizar acciones que afectan a líneas: filas, columnas y diagonales. En el modo 2D se pueden realizar acciones que afectan a la propia imagen o partes de la imagen.

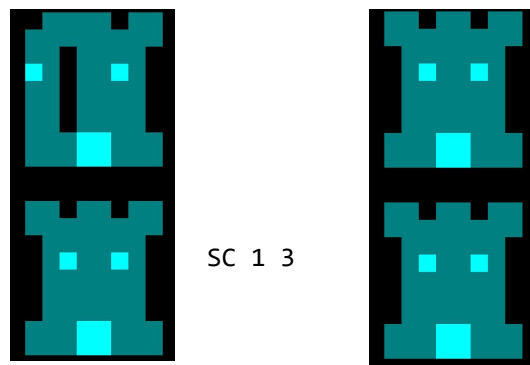


Figura 1

Figura 2

Se dispondrá de un conjunto de retos. Cada reto consta de dos imágenes (inicial y objetivo) y el número máximo de acciones que se pueden realizar para conseguirlo. Dicho reto se encuentra en un archivo. Por ejemplo, el archivo `torrel_1D.txt` corresponde a la Figura 1. Una vez cargado este fichero, se puede realizar la acción que intercambia las columnas 1 y 3 (SC 1 3). Esta acción deja la imagen de trabajo igual que el modelo objetivo como muestra la Figura 2, lo que finaliza este reto.

2. Descripción de la funcionalidad

Al comenzar el juego aparece un menú para seleccionar entre las versiones del juego modo 1D, modo 2D y la opción de salir. A continuación, se pide el nombre de fichero que contiene las imágenes inicial y objetivo. Además, el último elemento de este fichero es el número de acciones o intentos que se tienen para conseguir que la imagen inicial sea la imagen objetivo. Para comodidad del usuario, no es necesario introducir el nombre del fichero completo, pues cada reto se encuentra en un

archivo de nombre "nombre_XD.txt" donde X es el modo de juego (1 o 2), así que solo es necesario introducir por teclado el "nombre". Una vez cargado el reto, se mostrarán las imágenes (inicial y objetivo) y el número máximo de acciones que se pueden realizar. A continuación, se permitirá aplicar las acciones correspondientes al modo de juego, hasta que se consiga llegar al modelo objetivo o se realice el número máximo de acciones del reto. En ambos casos se mostrará un mensaje y se volverá al menú.

Las acciones en modo 1D son:

SF a b: intercambiar las filas **a** y **b** de la matriz.

SC a b: intercambiar las columnas **a** y **b** de la matriz.

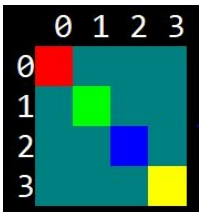
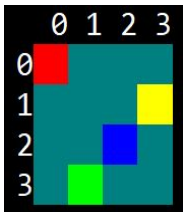
SD a: intercambiar las diagonales **a** y **-a**. Las diagonales positivas se numeran en los índices de las columnas, mientras que las diagonales negativas se numeran en los índices de las filas. La diagonal cero es la diagonal principal. Por ejemplo, la diagonal 1 está formada por los elementos [0,1], [1,2], [2,3], etc. mientras que la diagonal -1 está formada por los elementos [1,0], [2,1], [3,2], etc.

VF a: voltear la fila **a**. Dar la vuelta a la fila **a** con respecto al eje vertical central de la imagen.

VC a: voltear la columna **a**. Dar la vuelta a la columna **a** con respecto al eje horizontal central de la imagen.

VD a: voltear la diagonal **a**. Dar la vuelta a la diagonal **a** con respecto al centro de eje perpendicular de dicha diagonal.

Por ejemplo:

Acciones del modo 1D			
Imagen inicial	Fich. Acción	Significado	Resultado
	fichSF_1D.txt SF 1 3	intercambiar las filas 1 y 3	

	fichSC_1D.txt SC 1 3	intercambiar las columnas 1 y 3	
	fichSD_1D.txt SD 1	intercambiar las diagonales 1 y -1	
	fichVF_1D.txt VF 1	voltear la fila 1	
	fichVC_1D.txt VC 1	voltear la columna 1	
	fichVD_1D.txt VD 1	voltear la diagonal 1	

Las acciones en el modo 2D son:

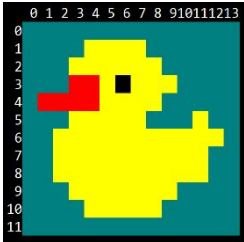
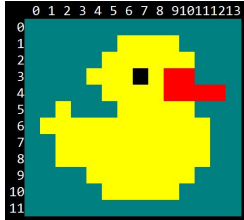
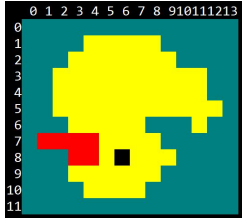
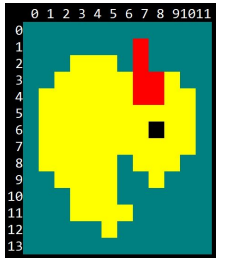

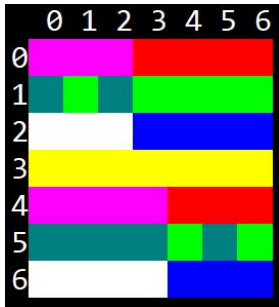
VV: dar la vuelta a la imagen completa con respecto a su eje vertical central.


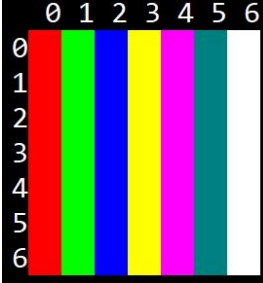
VH: dar la vuelta a la imagen completa con respecto a su eje horizontal central.

RD: rotar una imagen 90 grados en el sentido de las agujas del reloj.

SA a b c d: intercambiar las posiciones vecinas o adyacentes de las posiciones [a,b] y [c,d].

VD: Dar la vuelta a la imagen completa respecto a la diagonal principal

Acciones del modo 2D			
	Acción	Significado	Resultado
	patitaVV.txt VV	voltear respecto a la vertical	
	patitaVH.txt VH	voltear respecto a la horizontal	
	patitaRD.txt RD	rotar a la derecha la imagen	
	fichSA_2D.txt SA 1 1 5 5	intercambiar las posiciones vecinas de las posiciones {1,1} y {5,5}	

	fichVD_2D.txt VD	voltear respecto a la diagonal principal	
---	---------------------------------------	--	---

Las acciones que utilizan diagonales (VD y SD) requieren que la matriz sobre la que se aplican sea una matriz cuadrada. Las acciones sobre los elementos adyacentes, requiere que todos los elementos adyacentes estén contenidos en la matriz. En ambos casos, si se intenta aplicar la operación sobre una matriz que no cumpla los requisitos se considerará como un movimiento no válido.

3. Detalles de implementación

Una imagen consta de una matriz de dígitos¹ (`unsigned char`) que representan los índices de una paleta de 10 colores definida en el módulo `UtilidadesSYS` que se proporciona. Cada imagen tiene una resolución que establece el número de filas y de columnas de la matriz, ambas entre 0 y `DIM_MAX`. Define la constante `DIM_MAX` a 64, y la estructura `tMatrizChar` para representar las imágenes. La imagen no es necesariamente cuadrada. Define el tipo `tCoor` para estructurar la información de la posición de los elementos de la matriz. Debes definir cada tipo en su módulo: `Matriz` y `Coordenada`.

En el módulo `JuegoPM` se define el tipo `tJuegoPM` para estructurar los elementos del juego: las imágenes, el número máximo de acciones y el modo.

Añade también el módulo `UtilidadesSys` para las funciones dependientes del sistema operativo.

Formato de los archivos de entrada (retos)

Los archivos que contienen los retos (de nombre "`nombre_XD.txt`") empiezan con la imagen del modelo inicial, a continuación, la imagen objetivo y al final el número máximo de acciones:

```

NF NC           // Número de filas y columnas de la primera matriz (inicial)
D1 D2 ... DNC  // los dígitos de la primera fila
...            // NF - 2 filas de la primera matriz

```

¹ Los enteros positivos se pueden definir como: "`unsigned char`", "`unsigned short int`", "`unsigned int`", "`unsigned long int`" y "`unsigned long long int`". Cada tipo permite un rango de valores diferente.

```
...           // los dígitos de la última fila
NF NC         // Número de filas y columnas de la segunda matriz (objetivo)
D1 D2 ... DNC // los dígitos de la primera fila
...           // NF-2 filas de la segunda matriz
...           // los dígitos de la última fila
M             // número máximo de acciones del reto
```

Módulo Coordinada

Contendrá al menos el tipo de datos `tCoor` y las siguientes funciones de sobrecarga de operadores.

- `bool operator == (tCoor c1, tCoor c2):` compara si dos coordenadas son iguales.
- `bool operator != (tCoor c1, tCoor c2):` compara si dos coordenadas son distintas.
- `tCoor operator + (tCoor c1, tCoor c2):` devuelve la coordenada resultante de sumar las dos coordenadas dadas por parámetros.

Módulo Matriz

Contendrá al menos el tipo de datos `tMatrizChar` y las funciones

- `bool cargar(tMatrizChar & mat, istream & ent):` carga en la matriz los datos dados mediante el flujo de entrada. Devuelve falso si se produce un error en la lectura de los datos.
- `bool operator == (tMatrizChar const& mat1, tMatrizChar const& mat2):` compara si las dos matrices son iguales.
- `bool swap(tMatrizChar & mat, tCoor pos1, tCoor pos2):` intercambia las coordenadas pos1 y pos2 de la matriz. Devuelve falso si las posiciones no pertenecen al rango de la matriz.
- `bool swapF(tMatrizChar & mat, int f1, int f2):` intercambia las filas f1 y f2 de la matriz. Devuelve falso si las filas no pertenecen al rango de la matriz.
- `bool swapC(tMatrizChar & mat, int c1, int c2):` intercambia las columnas c1 y c2. Devuelve falso si las columnas no pertenecen al rango de la matriz.
- `bool swapD(tMatrizChar & mat, int d):` intercambia las diagonales d y -d. Devuelve falso si las diagonales no pertenecen al rango de la matriz, o si la matriz no es cuadrada.
- `bool voltearF(tMatrizChar & mat, int f):` dar la vuelta a la fila f con respecto al eje vertical central de la imagen. Devuelve falso si la fila no pertenece al rango de la matriz.
- `bool voltearC(tMatrizChar & mat, int c):` dar la vuelta a la columna c con respecto al eje horizontal central de la imagen. Devuelve falso si la columna no pertenece al rango de la matriz.

- **bool** voltearD(**tMatrizChar** & mat, **int** d): dar la vuelta a la diagonal d con respecto al centro de eje perpendicular de dicha diagonal. Devuelve falso si la diagonal no pertenece al rango de la matriz o si la matriz no es cuadrada.
- **void** voltearV(**tMatrizChar** & mat): dar la vuelta a la imagen completa con respecto a su eje vertical central.
- **void** voltearH(**tMatrizChar** & mat): dar la vuelta a la imagen completa con respecto a su eje horizontal central.
- **void** rotarD(**tMatrizChar** & mat): rotar una imagen 90 grados en el sentido de las agujas del reloj.
- **bool** swapAdy(**tMatrizChar** & mat, **tCoor** pos1, **tCoor** pos2): intercambiar las celdas vecinas o adyacentes de las posiciones pos1 y pos2.
- **bool** VoltearID(**tMatrizChar** & mat): Voltear la imagen completa respecto a la diagonal principal. Devuelve falso si la matriz no es cuadrada.

Módulo JuegoPM

Contendrá al menos el tipo de datos **tJuegoPM** y las funciones

- **void** mainJuegoPM(): función principal del juego.
- **int** menu(): menú para seleccionar el tipo de juego (1 o 2) o salir (0).
- **bool** iniciar(**tJuegoPM** & jpm & pm, **string** modo, **int** num): inicia los parámetros del juego que correspondan y llama a la función cargar definida a continuación.
- **bool** cargar(**tJuegoPM** & jpm): abre el fichero que corresponda y llama a la función cargar del módulo Matriz.
- **void** mostrar(**tJuegoPM** **const&** jpm) : muestra el estado del reto utilizando las facilidades del módulo utilidadesSYS. Debe mostrar primero la imagen que se está modificando y a continuación la imagen objetivo. Debe mostrar también el número de intentos que quedan.
- **bool** jugar(**tJuegoPM** & jpm): permite realizar las acciones necesarias para jugar y controla si se ha llegado al límite de acciones permitidas;
- **void** accion(**tJuegoPM** & jpm): según el comando de acción tecleado por el usuario, llama a la acción correspondiente definida en el módulo Matriz;

Módulo UtilidadesSys

Este módulo se proporciona a los alumnos. Contienen las funciones dependientes del sistema operativo para borrar la pantalla, manejar los colores de las imágenes etc.

4. Entrega de la práctica

La práctica se entregará en el Campus Virtual por medio de una tarea, que permitirá subir los archivos con extensiones *h* y *cpp* con el código fuente. Uno de los dos miembros del grupo será el encargado de subirlo, no lo suben los dos.

Recordad poner el nombre de los miembros del grupo en un comentario al principio de cada archivo del código fuente.

No se permite el uso de variables globales, ni el uso de instrucciones de salto: `goto`, `exit`, `continue`, `break` (salvo en instrucciones `switch`), ni `return` dentro de bucles.

Se valorará

- 1) La estructuración del código: instrucciones condicionales que cubran todos los casos, instrucciones iterativas que sigan los esquemas vistos en clase, definición y uso de constantes y funciones.
- 2) Documentación clara y concisa