# Introducción a los Computadores

***Curso 2016-2017***

# Práctica Final: Master Mind

El objetivo principal de esta práctica es crear un programa en lenguaje ensamblador del 8086 que implemente el juego *Master Mind*. La finalidad del juego es adivinar una combinación oculta formada por piezas de diferentes colores. Para ello, el jugador introduce, en cada jugada, una posible combinación y el programa responde indicándole cuántas piezas son correctas en posición y cuántas únicamente en color. Con esa información, el jugador vuelve a introducir una nueva combinación hasta adivinarla o superar el número de intentos.

## Especificaciones del programa

- Las posibles combinaciones ocultas estarán definidas en el propio programa como cadenas de hasta 6 piezas. Cada combinación tendrá asociado un identificador de juego de entre 5 posibles. Así, el rango de identificadores de juego estará comprendido entre 0 y 4.
- Al comenzar su ejecución, el programa mostrará una pantalla de inicio (figura 1) donde se le pedirá al usuario que introduzca el número de piezas que formarán la combinación oculta (valor entre 3 y 6) y el identificador del juego (valor entre 0 y 4).
- Tras obtener ambos datos, el programa mostrará el entorno del juego (figura 2) y esperará a que el usuario introduzca una tecla de acción o de juego.
- Las posibles teclas de acción y de juego serán las siguientes:
  - ‘I’: utilizada cuando el usuario quiere introducir una nueva combinación.
  - ‘N’: comienza un nuevo juego. El programa debe volver a mostrar la pantalla de inicio y pedir al usuario los datos indicados en el segundo punto (número de piezas e identificador de juego).
  - ‘S’: resuelve la combinación. Cuando el usuario introduzca esta tecla, el programa mostrará el código correcto y finalizará el juego actual (no el programa). En ese instante, quedará a la espera de que el usuario introduzca una nueva tecla de acción (‘N’ o ‘Esc’).
  - ‘Esc’(código de tecla 27): finaliza la ejecución del programa.
- Cuando el usuario introduzca la tecla ‘I’, el programa leerá las piezas de la combinación una a una. Cada pieza se asocia con un carácter de entre los siguientes: ‘R’ (rojo), ‘V’ (verde), ‘A’ (amarillo), ‘Z’ (azul), ‘B’ (blanco) y ‘M’ (marrón).
- Por cada pieza introducida, el programa comprobará que la pieza es válida (el carácter coincide con uno de los anteriores) y que no está repetida (la combinación no puede contener dos piezas del mismo color). Si se cumplen ambas condiciones, la pieza se mostrará en pantalla imprimiendo, en la posición que corresponda de la línea de intento actual, el carácter introducido del color asociado.
- Una vez que el usuario ha introducido la combinación completa, el programa comprobará cuántas piezas coinciden con la combinación oculta en posición y cuántas únicamente en color. Tras esto, mostrará en pantalla, en la línea de aciertos asociada con el intento actual, tantas V’s verdes como coincidencias existan en posición y, a continuación, tantas V’s azules como piezas se hayan acertado sólo en color. El resto de posiciones quedarán marcadas con una X en gris. Los aciertos no identifican las posiciones de las piezas correctas. Sólo informan al usuario sobre el número de piezas correctas en posición y/o color (figuras 2 y 3).
- Si la combinación completa fuera correcta, el programa mostrará la combinación oculta, así como un mensaje indicándole al usuario la situación (figura 4).
- El usuario dispondrá de 9 intentos para adivinar la combinación. Si tras dicho número de intentos la combinación introducida no fuera correcta, el programa no permitirá introducir nuevas combinaciones e informará al usuario de la situación, permitiéndole elegir una opción mediante las teclas de acción.

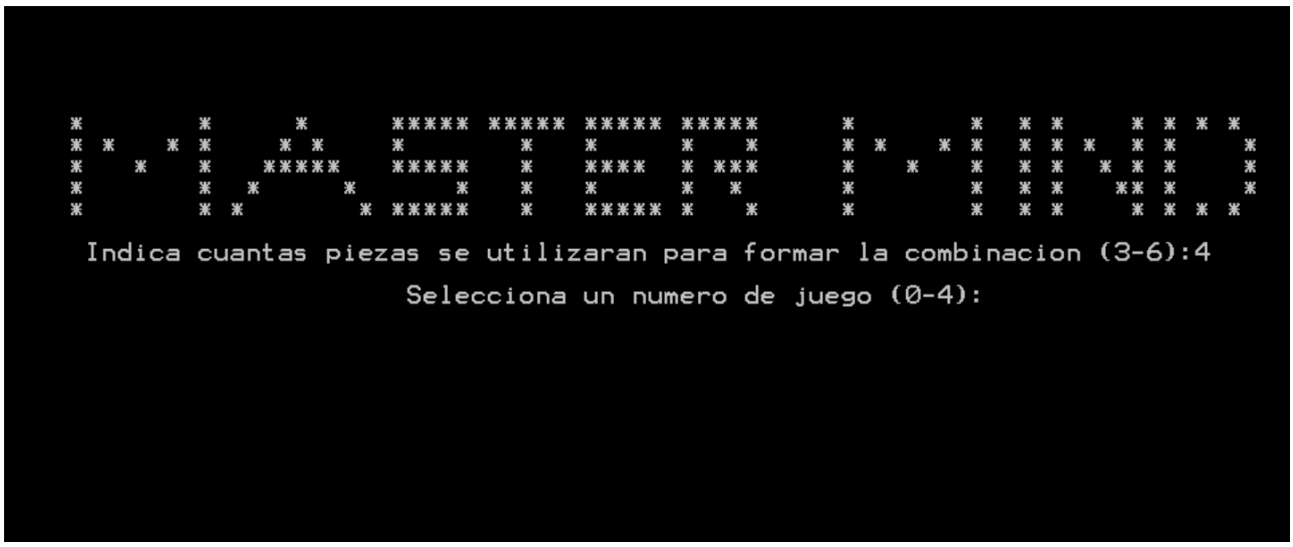


**Figura 1:** pantalla de inicio del juego.

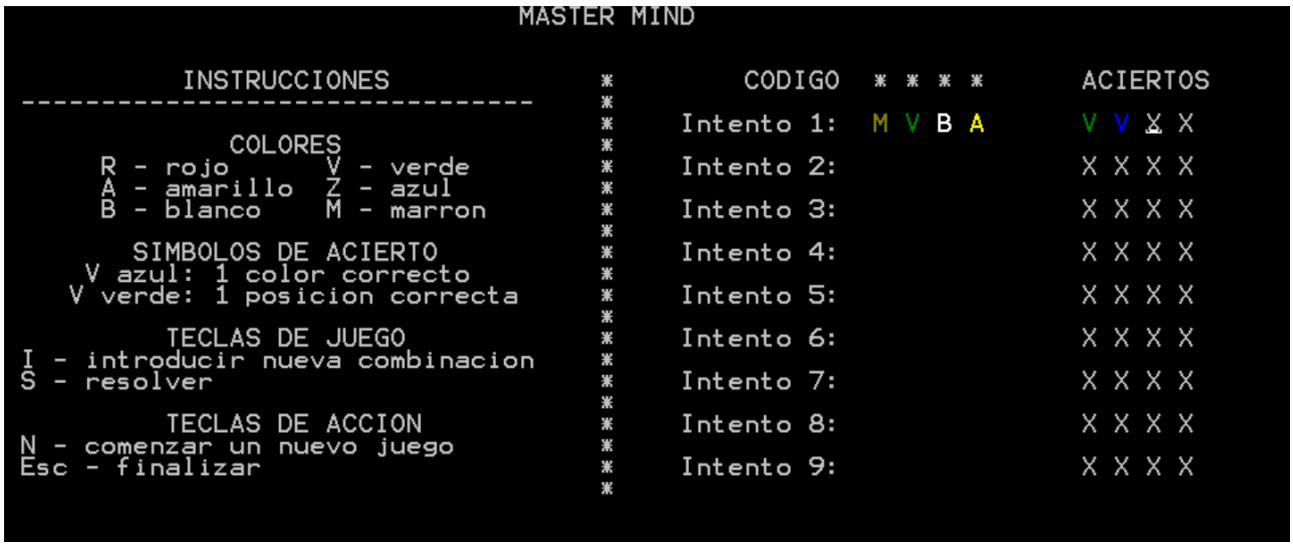


**Figura 2:** situación al inicio del juego (primer intento). Una pieza correcta en posición y color y otra sólo en color.

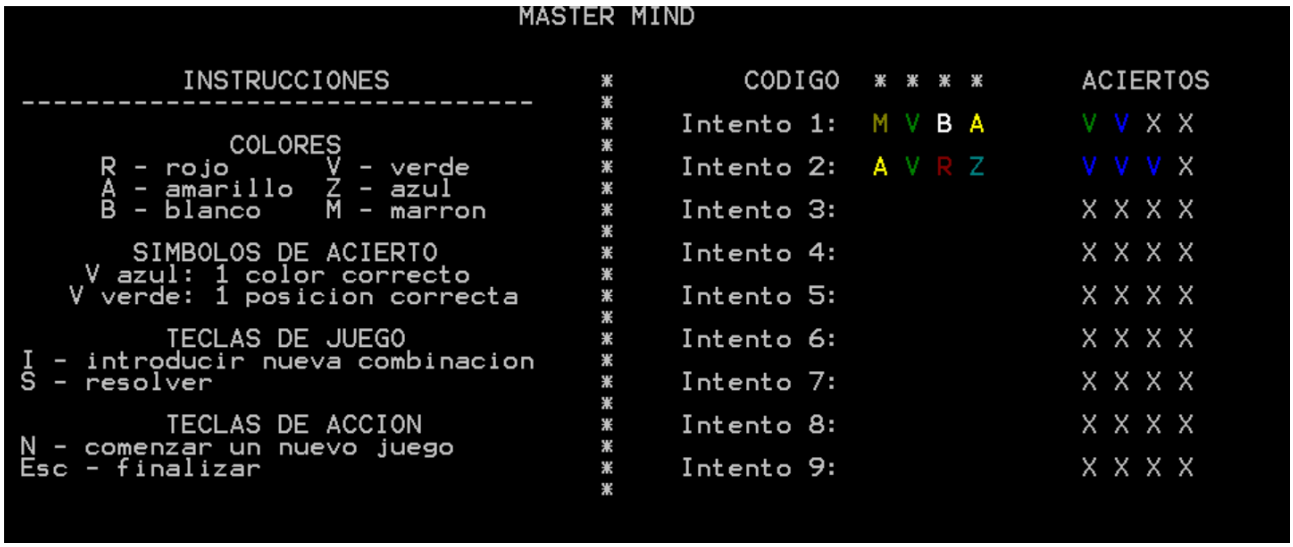


**Figura 3:** segundo intento de una jugada. 3 piezas correctas en color (no en posición)

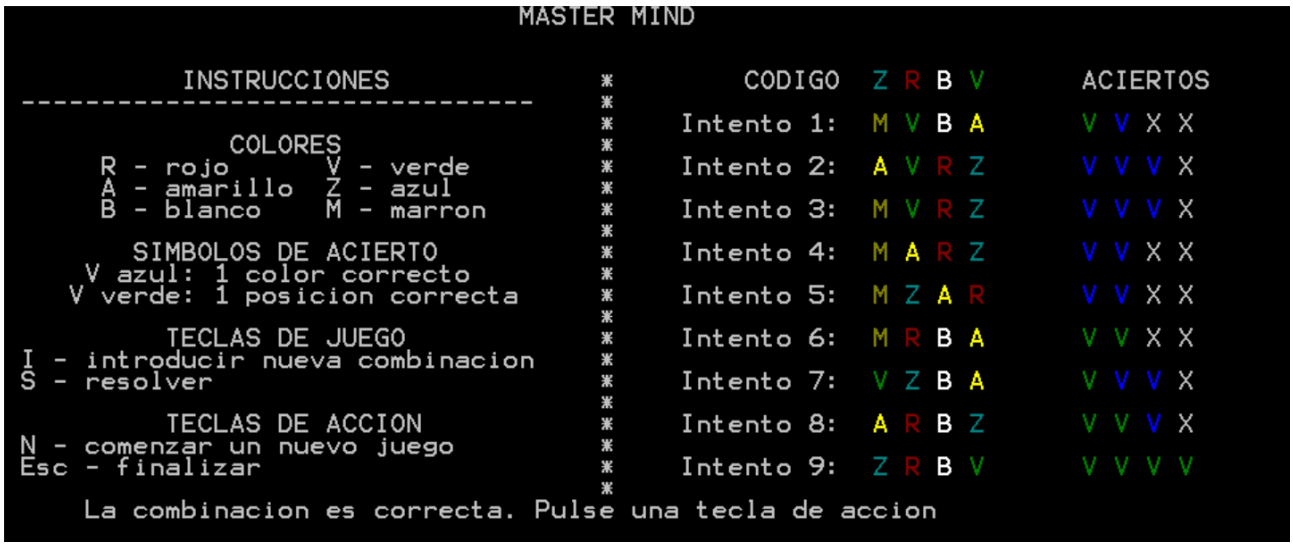


**Figura 4:** combinación correcta

Una ejecución del programa puede verse en: https://youtu.be/fzqgq03nOiA

## Objetivos de la práctica

El principal objetivo de la práctica es crear el código necesario en el fichero *mastermind.asm* para que el programa funcione según las especificaciones descritas anteriormente. En dicho fichero, se encuentra definida la estructura básica del programa, así como los datos necesarios para desarrollar el código.
El fichero (*mastermind.asm*) incluye el contenido de un segundo fichero (*entorno.inc*) donde se encuentran definidas constantes, variables y procedimientos que se usarán en el código a desarrollar. **El contenido de entorno.inc no debe modificarse**. Cada procedimiento incluido en este fichero está definido dentro de una macro cuyo nombre contiene el prefijo "DEFINIR_" y, a continuación, el nombre del procedimiento. Por ejemplo, el procedimiento "BorrarPantalla", encargado de borrar la pantalla completa, se encuentra definido dentro de la macro "DEFINIR_BorrarPantalla". Para invocar al procedimiento con la instrucción "call" es necesario indicar el nombre del procedimiento, no el de la macro. Así, en el ejemplo anterior, el uso del procedimiento BorrarPantalla se llevaría a cabo con la instrucción "call BorrarPantalla". La definición de variables sigue un sistema similar pero más simple.

En la siguiente sección se detallan cada una de las constantes, variables y procedimientos definidos en estos dos ficheros.

## Componentes del código fuente

A continuación, se detallan las constantes, variables y procedimientos que componen el código fuente en su estado inicial, todos se encuentran definidos en el fichero ***entorno.inc*** **que no debéis modificar en ningún caso**.

### Constantes

Todas las constantes incluidas definen posiciones de pantalla de los distintos elementos que componen la pantalla de inicio y la pantalla de juego. Las constantes que comienzan por el carácter "F" definen la fila de pantalla asociada con un elemento de la interfaz de usuario y, aquellas que empiezan por el carácter "C", la columna. Las siguientes figuras muestran la correspondencia entre cada pareja de constantes, formada por una fila y columna determinada, y el elemento de pantalla asociado.

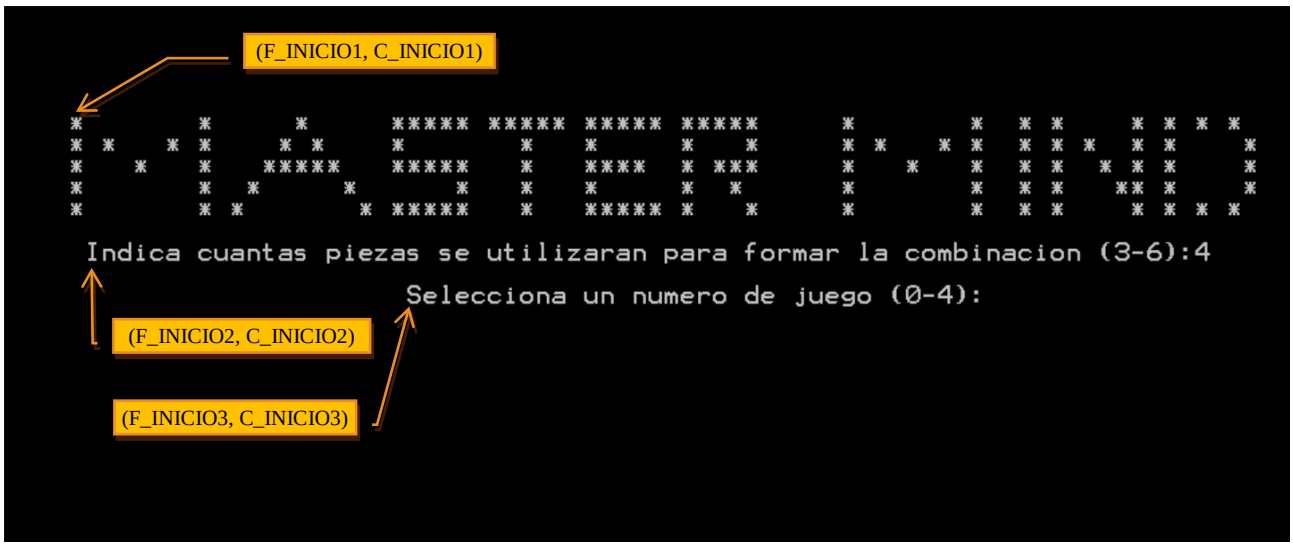


**Figura 5:** relación entre constantes y elementos de la pantalla de inicio

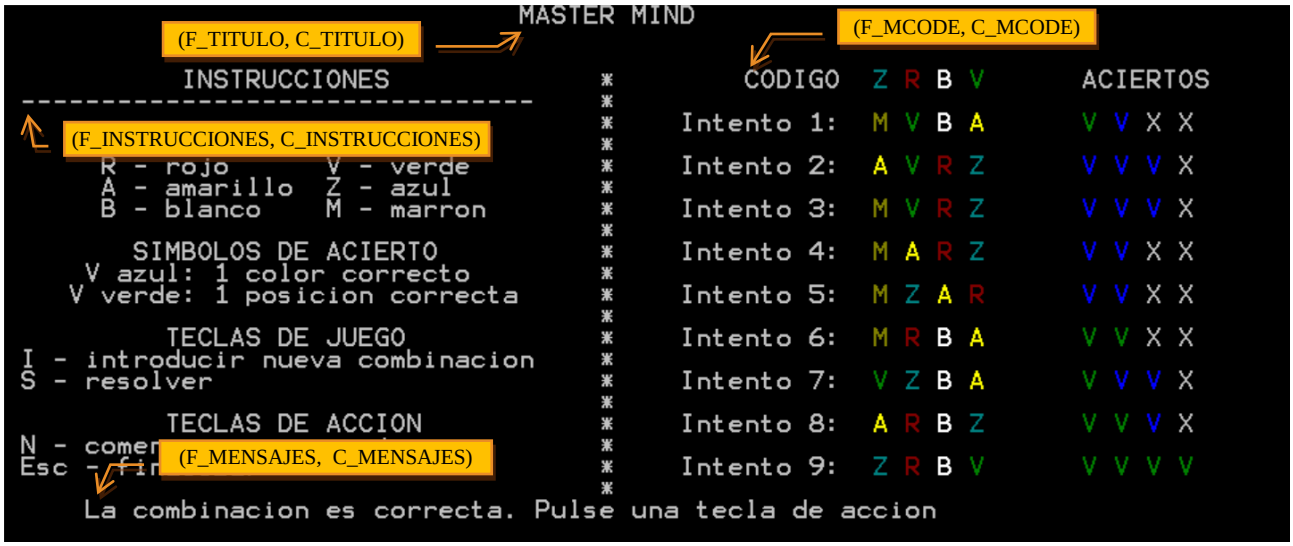


**Figura 6:** relación entre constantes y elementos de la pantalla de juego (1 de 2)

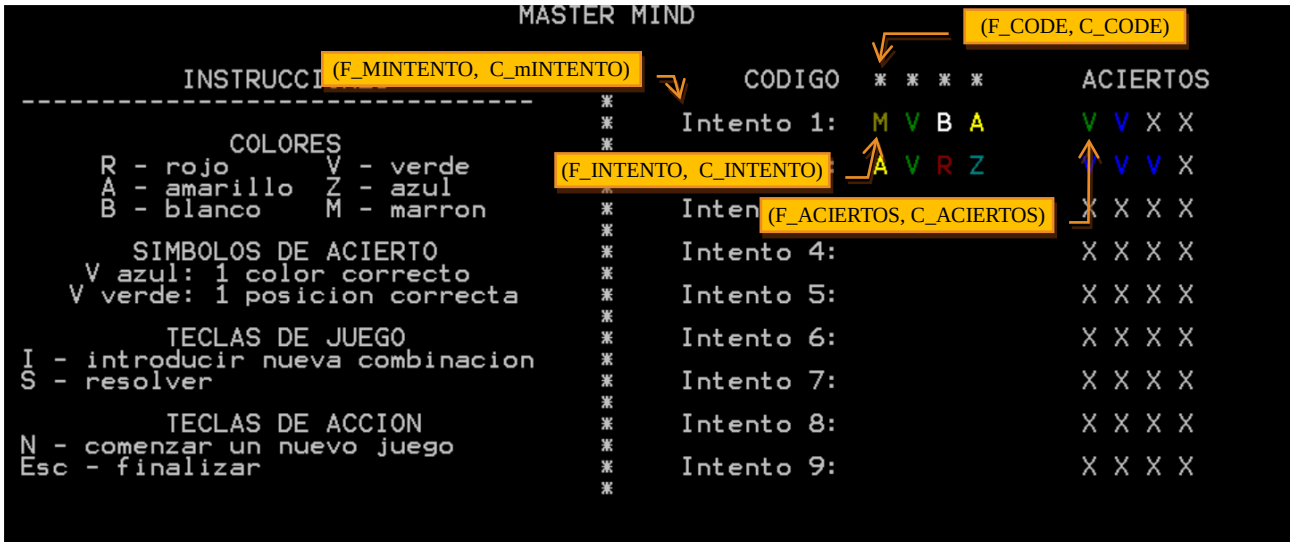


**Figura 6:** relación entre constantes y elementos de la pantalla de juego (2 de 2).

## Variables

- **fila**: fila de pantalla donde el procedimiento *ColocarCursor* sitúa el cursor.
- **colum**: columna de pantalla donde el procedimiento *ColocarCursor* sitúa el cursor.
- **caracter**: código de la tecla introducida por el usuario tras llamar al procedimiento *LeerTecla*.
- **d_inicio1**: contiene las líneas de texto para mostrar el dibujo del nombre del juego (“MASTER MIND”) en la pantalla de inicio. Las distintas líneas que componen el dibujo se imprimen utilizando una única llamada al procedimiento encargado de mostrar cadenas de texto (procedimiento “Imprimir”).
- **d_inicio2**: mensaje de petición del número de piezas de la combinación oculta. Debe mostrarse al usuario en la pantalla de inicio antes de leer por teclado dicho valor.
- **d_inicio3**: mensaje de petición del identificador de combinación oculta que se utilizará durante el juego. Debe mostrarse al usuario en la pantalla de inicio antes de leer por teclado dicho valor.
- **d_titulo**: contiene el título a mostrar en la parte superior de la pantalla de juego.
- **d_Mcode**: contiene la cabecera que se muestra sobre las líneas de intento de la pantalla de juego.
- **d_Mintento**: contiene el texto fijo de cada línea de intento.
- **d_Minstrucciones**: contiene las líneas de texto que se muestran en la parte de la izquierda de la pantalla de juego (instrucciones del programa).
- **msj_teclaAccion**: mensaje que debe mostrarse al usuario, en la posición de pantalla indicada por las constantes “F_MENSAJES” y “C_MENSAJES”, tras la ejecución de la opción “resolver” (tecla de acción ‘S’).
- **msj_superaIntentos**: mensaje que debe mostrarse al usuario, en la posición de pantalla indicada por las constantes “F_MENSAJES” y “C_MENSAJES”, cuando supere el número de intentos permitidos durante el desarrollo del juego.
- **msj_gana**: mensaje que debe mostrarse al usuario, en la posición de pantalla indicada por las constantes “F_MENSAJES” y “C_MENSAJES”, cuando adivine la combinación oculta.
- **npiezas**: número de piezas de la combinación oculta del juego actual. El programa debe inicializar esta variable, al inicio de cada nuevo juego, con el valor numérico indicado por el usuario en la petición correspondiente de la pantalla de inicio.
- **piezas_color**: códigos de color de las diferentes piezas. Esta variable se encuentra definida como un vector de tipo byte de 6 elementos, donde cada elemento contiene el color asociado a una pieza determina.
- **piezas_letra**: letras asociadas a las diferentes piezas. Al igual que la variable anterior, esta variable es un vector de 6 elementos, donde cada elemento contiene el carácter asociado con una pieza determinada. La relación entre carácter y color viene dada por la posición concreta dentro de estos dos vectores. Así, una pieza representada por la letra “piezas_letra[SI]” deberá imprimirse con el color indicado en “piezas_color[SI]”.
- **indJuego**: identificador del juego actual. El programa debe inicializar esta variable, al inicio de cada nuevo juego, con el valor numérico indicado por el usuario en la petición correspondiente de la pantalla de inicio.
- **juegos**: contiene 5 posibles combinaciones ocultas de hasta 6 piezas. La posición del elemento inicial de la combinación oculta que se está usando durante el juego actual se calcula multiplicando por 6 el contenido de la variable “indJuego”. Así, por ejemplo, la combinación oculta para el identificador de juego 2 se encuentra almacenada a partir de la posición 12 de la variable “juegos”. El número de elementos que forman la

combinación oculta a partir de esa posición inicial será el indicado por la variable "npiezas". Cada pieza contenida en esta variable está representada por la letra correspondiente.

- **combinacion**: vector de hasta 6 elementos donde se debe almacenar la combinación introducida por el usuario. Cada pieza dentro de este vector debe almacenarse con el carácter asociado para facilitar la comparación con la combinación oculta que corresponda de la variable "juegos".
- **aciertosPos**: número de piezas, de la combinación introducida por el usuario, correctas en posición y color.
- **aciertosColor**: número de piezas, de la combinación introducida por el usuario, correctas sólo en color. Tanto esta variable como la anterior deben actualizarse, cada vez que el usuario introduce una nueva combinación, como resultado de comparar la combinación introducida con la combinación oculta.
- **intento**: número de intentos utilizados por el usuario para el juego actual. Debe ponerse a 0 al inicio de un nuevo juego e incrementarse por cada combinación incorrecta introducida por el usuario.
- **finJuego**: variable booleana (0 ó 1) que permite identificar la situación de fin de juego. Esta variable tomará valor 1 cuando el usuario acierte la combinación, supere el número máximo de intentos permitidos o introduzca la opción "resolver juego" (tecla de acción 'S').

**Procedimientos**

- **BorrarPantalla**: borra la pantalla completa. Debe ser invocado antes de mostrar la pantalla de inicio (como resultado de iniciar un nuevo juego).
- **ColocarCursor**: coloca el cursor en la posición indicada por las variables "fila" y "colum".
- **Imprimir**: muestra por pantalla, en la posición actual del cursor, la cadena de texto direccionada por el registro DX.
- **LeerTecla**: lee un carácter de teclado, sin salida por pantalla, y lo devuelve en la variable "caracter".
- **ImprimeCartacerColor**: imprime, en la posición actual del cursor, el carácter indicado en el registro AL con el color especificado en el registro BL.
- **DibujarIntentos**: imprime las líneas de intento en la pantalla de juego. Este procedimiento es utilizado por el procedimiento "DibujarEntorno".
- **DibujarCodigo**: imprime la cabecera de las líneas de intento. Procedimiento utilizado por "DibujarEntorno".
- **DibujarInstrucciones**: imprime el mensaje completo con las instrucciones del juego. Procedimiento utilizado por "DibujarEntorno".
- **DibujarEntorno**: muestra la pantalla de juego al completo utilizando los 3 procedimientos anteriores. Este procedimiento debe ser invocado cuando comienza un nuevo juego.
- **MuestraAciertos**: muestra los aciertos especificados en las variables "aciertosPos" y "aciertosColor" en la línea de intento indicada por la variable "intento". Debe invocarse tras comparar la combinación introducida por el usuario con la combinación oculta y almacenar el número de aciertos de posición y/o color en las variables correspondientes ("aciertosPos" y "aciertosColor").
- **MuestraCombinacion**: muestra la combinación oculta en la posición correspondiente de la pantalla de juego. Debe invocarse cuando el usuario seleccione la opción "resolver juego".
- **MuestraGana**: imprime, en la posición correspondiente de la pantalla de juego, el mensaje que le indica al usuario que ha acertado la combinación oculta (contenido de la variable "msj_gana"). Además, muestra dicha combinación llamando al procedimiento anterior. Este procedimiento debe invocarse una vez que se detecta que el usuario ha acertado la combinación oculta.