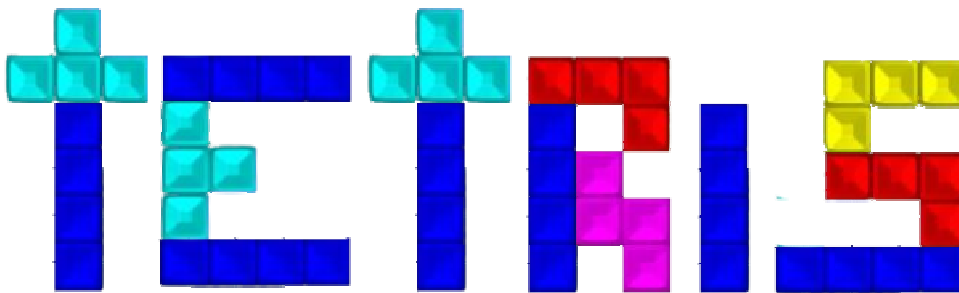


# Estructura y Tecnología de Computadores

---

## PRÁCTICA DE PROGRAMACIÓN EN ENSAMBLADOR MIPS R2000 CON EL SIMULADOR PCSPIM 1.0



**Componentes:** Fernández Candel, Alejandro Tomás (DNI: 77718649-Q)  
Fernández Vivancos, Javier (DNI: 15482513-V)  
**Titulación, Grupo:** I. Técnica en Informática de Sistemas, Grupo 1  
**Profesor:** Juan José Vera Guirao

# Índice

- 1. Implementación**
  - 1.1. Segmento de datos (.data)**
    - 1.1.1. Tablero**
    - 1.1.2. Piezas**
    - 1.1.3. Variables Globales**
  - 1.2. Segmento de texto (.text)**
    - 1.2.1. Rutinas**
      - 1.2.1.1. Direccion\_celda**
      - 1.2.1.2. Obten\_celda**
      - 1.2.1.3. Cambia\_celda**
      - 1.2.1.4. Imprime\_tablero**
        - Imprime\_fila
        - Imprime\_sombra
      - 1.2.1.5. Inicializa\_tablero**
      - 1.2.1.6. Limpiar\_fila**
        - Fila\_llena
        - Limpiar\_fila\_llena
      - 1.2.1.7. Copia\_matriz**
      - 1.2.1.8. Direccion\_pieza**
      - 1.2.1.9. Coloca\_pieza**
      - 1.2.1.10. Genera\_prov**
      - 1.2.1.11. Fija\_actual**
      - 1.2.1.12. Act\_nueva\_pieza**
      - 1.2.1.13. Act\_pos\_prov**
      - 1.2.1.14. Configura\_tablero**
      - 1.2.1.15. Num\_aleatorio**
      - 1.2.1.16. Imprime\_opciones**
      - 1.2.1.17. Act\_puntuacion**
      - 1.2.1.18. Útiles**
        - Leer\_teclado\_int
        - Impr\_string
        - Impr\_int
    - 1.2.2. Programa principal (main)**
  - 1.3. Mejoras**
    - 1.3.1. Tablero tamaño variable**
    - 1.3.2. Sistema de puntuación**
    - 1.3.3. Diseño de piezas nuevas**
    - 1.3.4. Pieza siguiente a la derecha del tablero**
    - 1.3.5. Sombra de la pieza en el tablero**
    - 1.3.6. Giro de las piezas (0°, 90°, 180° y 270°)**
    - 1.3.7. Opción 'bajar hasta colocar'**

# 1. Implementación

## 1.1. Segmento de datos (.data)

### 1.1.1. Tablero

Existen tres matrices distintas para el tablero **tablero**, **tablero\_print** y **tablero\_prov** cada matriz tiene un tamaño de 1000 bytes (tamaño máximo establecido para el tablero). Cada tablero tiene un uso específico:

- **tablero**: tablero original donde se encuentra todas las celdas ocupadas (de forma fija) del tablero en juego.
- **tablero\_print**: tablero que solamente se usará para imprimir por consola el tablero en juego.
- **tablero\_prov**: tablero que se usa para generar los posibles movimientos las piezas que el usuario pida.

En cada celda de la matriz de los tableros se guardará un código para indicar los tres posibles estados de una celda:

- **celda vacía = -2** (celda en blanco).
- **celda ocupada = -3** (celda ocupada de forma fija).
- **celda actual = -1** (celda ocupada por la pieza en juego).

Existen también unas variables donde guardaremos el tamaño del tablero, altura y anchura que el usuario introduzca a inicio del juego:

- **tam\_tablero\_x**: anchura del tablero.
- **tam\_tablero\_y**: altura del tablero.

### 1.1.2. Piezas

Por cada pieza del juego existen cuatro **matrices de 16 bytes**, una matriz para cada posible giro:  $0^\circ$ ,  $90^\circ$ ,  $180^\circ$  y  $270^\circ$  donde en cada celda de la matriz guardará un código para saber que celdas ocupa dicha pieza que en este caso contendrán un 1 y las celdas vacías contendrán un 0. Las matrices de las piezas son de orden cuatro (4x4) y para conocer el tamaño de la pieza disponemos de dos vectores, uno para cada dimensión de la pieza (altura y anchura):

- **piezas\_x**: anchura de la pieza.
- **piezas\_y**: anchura de la pieza.

En relación con las piezas existen variables donde guardaremos información sobre la pieza en juego y también sobre la siguiente pieza a jugar. Para conocer el número de piezas que tenemos en el juego existe la variable **num\_piezas** que guarda el número de piezas disponibles.





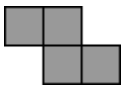
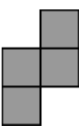
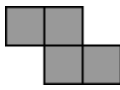

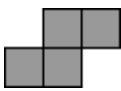
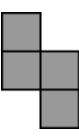
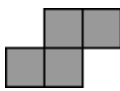

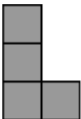
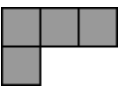

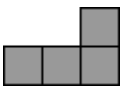
Información de la pieza en juego:

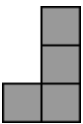




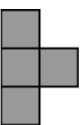

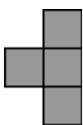






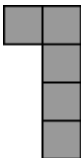

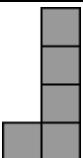







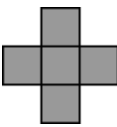
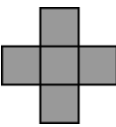
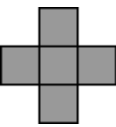
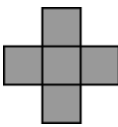
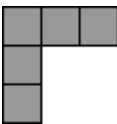
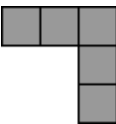

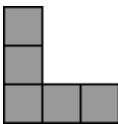
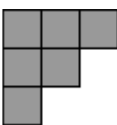
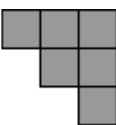

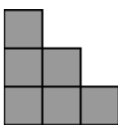




- **pieza\_en\_juego**: número de la pieza en juego.
- **giro\_en\_juego**: giro de la pieza en juego.
- **pos\_pieza\_x**: posición X de la pieza en juego.
- **pos\_pieza\_y**: posición Y de la pieza en juego.
- **pos\_prov\_pieza\_x**: posición X provisional de la pieza en juego.
- **pos\_prov\_pieza\_y**: posición Y provisional de la pieza en juego.
- **giro\_prov\_pieza**: giro provisional de la pieza en juego.

Otra información:

- **pieza\_siguiente**: número de la siguiente pieza a jugar.
- **orden\_matriz\_pieza**: orden de la matriz de la pieza.
- **tam\_matriz\_pieza**: tamaño de la matriz.

El diseño de las piezas son los siguientes:

	$0^\circ$	$90^\circ$	$180^\circ$	$270^\circ$
Pieza 0				
Pieza 1				
Pieza 2				
Pieza 3				

Pieza 4				
Pieza 5				
Pieza 6				
Pieza 7				
Pieza 8				
Pieza 9				
Pieza 10				
Pieza 11				
Pieza 12				
Pieza 13				

### 1.1.3. Variables Globales

En el segmento de datos encontramos también mensajes que usaremos pedir al usuario información/acción que quiere hacer así como también indicar información como su puntuación y fin del juego.

Existen otras variables como **puntuacion\_total** que guarda la puntuación que tiene el usuario y una variable que usaremos para imprimir por consola cada celda del tablero **celda\_impr**.

## 1.2. Segmento de texto (.text)

### 1.2.1. Rutinas

#### 1.2.1.1. Direccion\_celda

La rutina **Direccion\_celda** tiene como función calcular la dirección de memoria de una celda del tablero a partir de la posición de la celda (x,y). La rutina tiene los siguiente parámetros de entrada: **registro \$a0** (dirección de la matriz), **registro \$a1** (fila de la celda), **registro \$a2** (columna de la celda) y **registro \$a3** (tamaño de la fila de la matriz). Y como parámetro de salida el **registro \$v0** (dirección de memoria de la celda (\$a1, \$a2)).

Nota: esta rutina será la única forma de conseguir la dirección de memoria de una celda.

#### 1.2.1.2. Obten\_celda

La rutina **Obten\_celda** tiene como función la obtención del código de la celda (estado en el que se encuentra: vacía, ocupada o actual). La rutina tiene los siguientes parámetros de entrada: **registro \$a0** (dirección del tablero), **registro \$a1** (fila de la celda), **registro \$a2** (columna de la celda) y **registro \$a3** (tamaño de la fila de la matriz). Y como parámetro de salida: **registro \$v0** (el valor que contiene la celda (\$a1, \$a2)).

Nota: esta rutina será la única forma de conseguir el contenido de la celda.

#### 1.2.1.3. Cambia\_celda

La rutina **Cambia\_celda** tiene como función cambiar el código de la celda (estado en el que se encuentra: vacía, ocupada o actual). La rutina tiene los siguientes parámetros de entrada: **registro \$a0** (dirección del tablero), **registro \$a1** (fila de la celda), **registro \$a2** (columna de la celda) y **registro \$a3** (valor a poner en la celda). No tiene parámetros de salida.

Nota: esta rutina será la única forma de cambiar el código de las celdas.

#### 1.2.1.4. Imprime\_tablero

La rutina **Imprime\_tablero** tiene como función imprimir por consola un conjunto de elementos: el tablero, la pieza siguiente (a la derecha del tablero), la puntuación y la sombra de la pieza en juego al pie del tablero. Esta rutina está dividida en tres módulos, tenemos la rutina principal **Imprime\_tablero** y dos subrutinas **Imprime\_fila** e **Imprime\_sombra**. De forma que la impresión por consola se realiza por filas con la rutina **Imprime\_fila**. La rutina **Imprime\_sombra** es la encargada de imprimir la sombra de la pieza al pie del tablero.

La rutina **Imprime\_tablero** tiene los siguiente parámetros de entrada: **registro \$a0** (dirección del tablero), **registro \$a1** (tamaño de la fila) y **registro \$a2** (altura del tablero). No tiene parámetros de salida.

La rutina **Imprime\_fila** tiene los siguiente parámetros de entrada: **registro \$a0** (dirección del tablero), **registro \$a1** (fila) y **registro \$a2** (tamaño de la fila). No tiene parámetros de salida.

La rutina **Imprime\_sombra** tiene los siguiente parámetros de entrada: **registro \$a0** (posición x de la pieza) y **registro \$a1** (anchura de la pieza). No tiene parámetros de salida.

#### 1.2.1.5. Inicializa\_tablero

La rutina **Inicializa\_tablero** tiene como función inicializar todas las celdas de un tablero con un valor (código vacío, ocupado o actual). La rutina **Inicializa\_tablero** tiene los siguiente parámetros de entrada: **registro \$a0** (dirección del tablero), **registro \$a1** (valor con el que inicializar), **registro \$a2** (tamaño de la fila) y **registro \$a3** (altura del tablero). No tiene parámetros de salida.

#### 1.2.1.6. Limpiar\_fila

La rutina **Limpiar\_fila** tiene como función buscar filas llenas por completo eliminarlas, copiar todas las filas superiores y calcular los puntos para la puntuación en función del número de filas eliminadas. Esta rutina está dividida en tres módulos, una rutina principal **Limpiar\_fila** y dos subrutinas **Fila\_llena** y **Limpiar\_fila\_llena**. De forma que la rutina **Fila\_llena** compraba si una fila está llena por completo y la rutina **Limpiar\_fila\_llena** es la encargada de ir copiando las filas superiores a partir de la fila llena eliminando así la fila llena.

La rutina **Limpiar\_fila** tiene el siguiente parámetro de entrada: **registro \$a0** (dirección del tablero). No tiene parámetros de salida.

La rutina **Fila\_llena** tiene los siguientes parámetros de entrada: **registro \$a0** (dirección del tablero), **registro \$a1** (número de la fila) y **registro \$a2** (tamaño de la fila). Y como parámetro de salida: **registro \$v0** (0 si la fila está llena, -1 si no lo está).

La rutina **Limpiar\_fila\_llena** tiene los siguientes parámetros de entrada: **registro \$a0** (dirección del tablero) y **registro \$a1** (número de la fila llena). No tiene parámetros de salida.

#### 1.2.1.7. Copia\_matriz

La rutina **Copia\_matriz** tiene como función copiar un tablero sobre otro tablero.

La rutina **Copia\_matriz** tiene los siguientes parámetros de entrada: **registro \$a0** (dirección del tablero a copiar) y **registro \$a1** (dirección del tablero donde guarda la copia). No tiene parámetros de salida.

#### 1.2.1.8. Direccion\_pieza

La rutina **Dirección\_pieza** tiene como función calcular la dirección de memoria de una pieza teniendo en cuenta la pieza y el giro, también calcula la posición dentro del vector donde se encuentra el tamaño de la pieza (vectores **piezas\_x** y **piezas\_y**).

La rutina **Dirección\_pieza** tiene los siguientes parámetros de entrada: **registro \$a0** (número de la pieza y registro) y **registro \$a1** (número de giro). Y como parámetros de salida: **registro \$v0** (dirección de memoria de la pieza) y **registro \$v1** (posición dentro del vector del tamaño de la pieza).

#### 1.2.1.9. Coloca\_pieza

La rutina **Coloca\_pieza** tiene como función colocar la pieza en un tablero teniendo como posibles resultados su correcta o incorrecta colocación.

La rutina **Coloca\_pieza** tiene los siguientes parámetros de entrada: **registro \$a0** (dirección del tablero), **registro \$a1** (número de la pieza) y **registro \$a2** (número de giro). Y como parámetros de salida: **registro \$v0** (0 si se coloca la pieza correctamente, -1 si no se coloca la pieza).



#### 1.2.1.10. Genera\_prov

La rutina **Genera\_prov** tiene como función generar un tablero realizando el movimiento que el usuario introduzca y en el caso de que el movimiento sea posible lo realizará en caso contrario no realizará ningún cambio. Esta rutina utiliza rutinas como **Copia\_matriz** y **Coloca\_pieza** como son rutinas que pueden usarse fuera de la rutina **Genera\_prov** no las consideramos subrutinas de la rutina **Genera\_prov**.

La rutina **Genera\_prov** tiene los siguientes parámetros de entrada: **registro \$a0** (dirección tablero), **registro \$a1** (dirección tablero\_prov) y **registro \$a2** (dirección tablero\_print). Y como parámetros de salida: **registro \$v0** (0 si tablero\_prov se ha generado correctamente, -1 si no se ha generado correctamente).

#### 1.2.1.11. Fija\_actual

La rutina **Fija\_actual** tiene como función cambiar las celdas que contengan el código 'actual' por el código 'ocupado', de forma que la pieza en juego pasa a ser fija.

La rutina **Fija\_actual** tiene el siguiente parámetro de entrada: **registro \$a0** (dirección del tablero). No tiene parámetros de salida.

#### 1.2.1.12. Act\_nueva\_pieza

La rutina **Act\_nueva\_pieza** tiene como función actualizar/preparar las variables globales para poner una nueva pieza en juego. Poniendo la posición de la pieza en la primera fila y en el centro del tablero y el giro inicial a 0 grados.

La rutina **Act\_nueva\_pieza** tiene los siguientes parámetros de entrada: **registro \$a0** (número de la pieza) y **registro \$a1** (tamaño de la fila del tablero). No tiene parámetros de salida.

#### 1.2.1.13. Act\_pos\_prov

La rutina **Act\_pos\_prov** tiene como función actualizar/copiar la posición de la pieza a las variable de la posiciones provisionales.

La rutina **Act\_pos\_prov** tiene los siguientes parámetros de entrada: **registro \$a0** (posición de memoria de la posición X de la pieza), **registro \$a1** (posición de memoria de la posición Y de la pieza) y **registro \$a2** (Posición de memoria del giro). No tiene parámetros de salida.

#### 1.2.1.14. Configura\_tablero

La rutina **Configura\_tablero** tiene como función comprobar que el tamaño del tablero sea el correcto, tamaño mínimo 16 celdas (matriz de las piezas) y el tamaño máximo 1000 celdas (memoria reservada para el tablero).

La rutina **Configura\_tablero** tiene los siguientes parámetros de entrada: **registro \$a0** (anchura del tablero) y **registro \$a1** (altura del tablero). Y como parámetro de salida: **registro \$v0** (0 si el tamaño es posible, -1 si el tamaño no es válido).

#### 1.2.1.15. Num\_aleatorio

La rutina **Num\_aleatorio** tiene como función generar un número aleatorio entre 0 y una cota superior que se introduce como parámetro.

La rutina **Num\_aleatorio** tiene el siguiente parámetro de entrada: **registro \$a0** (cota superior para el número aleatorio). Y como parámetro de salida: **registro \$v0** (Número aleatorio generado).

#### 1.2.1.16. Imprime\_opciones

La rutina **Imprime\_opciones** tiene como función imprimir las cadenas del menú de opciones por consola. No tiene parámetros de entrada o salida.

#### 1.2.1.17. Act\_puntuacion

La rutina **Act\_puntuacion** tiene como función añadir puntos a la puntuación total en juego.

La rutina **Act\_puntuacion** tiene el siguiente parámetro de entrada: **registro \$a0** (puntos a añadir). No tiene parámetros de salida.

#### 1.2.1.18. Útiles

Las siguientes rutinas: **Leer\_teclado\_int**, **Impr\_string** y **Impr\_int** se han añadido para facilitar la lectura de enteros por teclado y la impresión por consola de cadenas de texto y enteros.

La rutina **Leer\_teclado\_int** sólo tiene como parámetro de salida: **registro \$v0** (entero leído desde teclado). Las rutinas **Impr\_string** y **Impr\_int** sólo como parámetro de entrada: **registro \$a0** (Dirección de la cadena o entero a imprimir por consola).

## 1.2.2. Programa principal (main)

En la sección main del programa encontramos el código encargado de realizar las operaciones necesarias para realizar las opciones introducidas por el usuario. Lo primero que se realiza es pedir al usuario el tamaño del tablero, tras comprobar que el tamaño es válido inicializamos el tablero vacío. Luego generamos el número de una pieza (será la primera pieza en salir). A continuación iniciamos los valores necesarios para poner la pieza en el lugar de salida y generamos un número para la pieza siguiente e imprime por consola el tablero con la pieza en juego y el menú de opciones, pedimos al usuario la acción a realizar y después intentamos generar la acción elegida por el usuario si es posible actualizamos las variables necesarias (posición de la pieza, giro) y se realizan operaciones necesarias teniendo en cuenta las posibles situaciones que podemos encontrar:

- La pieza puede seguir moviéndose: volvemos a imprimir el tablero y a pedir la opción al usuario.
- Es necesario fijar la pieza: fijamos la pieza, tenemos que poner la pieza siguiente como la nueva pieza en juego y generar otra pieza siguiente, reiniciar los valores de la posición de la pieza y giro a los iniciales, imprimir el tablero y decir la opción al usuario.
- La pieza no puede colocarse porque el tablero está demasiado lleno: el juego termina.
- El usuario elige la opción salir: el juego termina.

## 1.3. Mejoras

Las mejoras realizadas en la practica son las siguientes: tablero tamaño variable, sistema de puntuación, diseño de piezas nuevas, mostrar pieza siguiente a la derecha del tablero, sombra de la pieza en el tablero, giro de las piezas (0°, 90°, 180° y 270°) y opción 'bajar hasta colocar'. A continuación explicaremos cada una de estas mejoras como sus aspectos relevantes.

### 1.3.1. Tablero tamaño variable

Al inicio del programa se pide al usuario que introduzca el tamaño del tablero, (anchura y altura) teniendo como límite un tablero mínimo de 16 celdas (tamaño de la matriz de las piezas) y máximo de 1000 celdas (memoria reservada para el tablero). Para comprobar si el tamaño que introduce el usuario es válido hemos añadido una rutina llamada **Configura\_tablero** (arriba comentada).

### 1.3.2. Sistema de puntuación

El sistema de puntuación se inicia a 0 al comenzar a jugar y la forma de conseguir puntos es por piezas jugadas y filas eliminadas (por pieza jugada 1 punto y por fila eliminada se dan los puntos correspondientes al número de celda de la fila, es decir, la anchura del tablero). Para implementar esta mejora hemos añadido una variable global **puntuacion\_total** (donde se guarda la puntuación total), una cadena de texto y una rutina llamada **Act\_puntuacion** que es la encargada de ir aumentando la puntuación añadiendo los puntos que le pasemos como parámetro. La puntuación se imprime por consola junto con el tablero y la siguiente pieza cuando se hace una llamada a la rutina **Imprime\_tablero**.

### 1.3.3. Diseño de piezas nuevas

Se han añadido nuevos diseños de piezas (exactamente 7 piezas) teniendo un total de 14 piezas (las piezas originales y las piezas añadidas).

### 1.3.4. Pieza siguiente a la derecha del tablero

En la rutina **Imprime\_tablero** se ha añadido la impresión de la siguiente pieza que saldrá tras colocar la pieza en juego. Se ha añadido la variable **pieza\_siguiente** donde se guarda el número de la pieza.

### 1.3.5. Sombra de la pieza en el tablero

Hemos añadido una subrutina llamada **Imprime\_sombra** que utilizaremos en la rutina **Imprime\_tablero**, que imprimirá al pie del tablero el lugar donde caería la pieza si se realizaran todos los movimientos hacia abajo.

### 1.3.6. Giro de las piezas (0°, 90°, 180° y 270°)

Se ha implementado el juego de forma que cada pieza puede girarse en giros de 90 grados. Por cada pieza se han añadido 3 matrices donde se guarda cada giro posible, teniendo 4 matrices por pieza donde cada una es para cada giro 0°, 90°, 180° y 270°. Se ha añadido una variable **giro\_en\_juego** donde se guarda el giro de la pieza en juego. Las piezas aparecen con un giro inicial correspondiente a 0 grados.

### **1.3.7. Opción ‘bajar hasta colocar’**

En el menú de opciones hemos añadido una opción que con la función de ‘bajar hasta colocar’, de forma que al usar dicha opción bajará la pieza hasta el lugar que corresponda y fijándola en el lugar donde corresponda. La implementación de esta opción se ha hecho dentro de la opción bajar de forma que al comprobar que la opción elegida es ‘bajar hasta colocar’ realiza un bucle hasta que coloca la pieza como fija.