

Parte Práctica: Tetris

En ISW-Games siguen incursionando en la creación de juegos. El prototipo anterior del Pacman satisfizo a los product owners y quieren hacer algo parecido pero con el juego del **Tetris**. O sea, la idea es implementar un prototipo simplificado del **Tetris** para ver cuánto costaría hacerlo con **TDD** y si tendría sentido empezar a comercializarlo.


Lo que se espera para asegurar el correcto funcionamiento del juego es utilizar una representación string de cómo quedaría el tablero luego de ejercitarlo en cada test. Por ejemplo:

```
'#---***---#'  
'#---*-----#'  
'#-----#'  
'#---**---#'  
'#---**---#'  
'#####'
```

Esta es la representación string de un juego de **Tetris** que consta de 5 filas por 10 columnas, donde la pieza con forma de Cuadrado (Tetromino_O) llegó a la última fila y está saliendo otra pieza con forma de T (Tetromino_T). Se denomina Tetromino a una forma geométrica compuesta de cuatro cuadrados iguales


A diferencia del Pacman, no es necesario crear el juego con un tablero inicial definido con strings. En este caso el juego debe ser instanciado mínimamente con la extensión del tablero (columnas x filas), la cual debe tener como mínimo 4 columnas y 5 filas.


Las piezas del **Tetris** que hay que implementar son:

1. **Tetromino O**: Es la pieza que representa un cuadrado de 2x2: 
2. **Tetromino T**: Es la pieza que representa una T, con 3 cuadrados arriba y un

cuadrado abajo en el medio: 

3. **Tetromino I**: Es la pieza que es como un I, de 4 cuadrados: 

4. **Tetromino Z**: Es la pieza con 2 cuadros arriba y 2 abajo corridos a derecha: 

5. **Tetromino L**: Es la pieza con forma de L: 

En la representación String del tablero, las piezas deben ser presentadas por medio de asteriscos (*), los espacios libres por medio de guiones (-) y la pared por medio de hashtag (#).

El juego del **Tetris** debe permitir las siguientes **acciones**:

1. Que el juego avance. Para ello debe haber un mensaje, por ejemplo *#tick*, que le indique al juego que debe mover hacia abajo una fila la pieza que se está jugando. Solo puede haber una pieza en juego
2. Indicarle a la pieza actual que se mueva a izquierda o derecha. Se puede mover varias veces entre *#tick* y *#tick*. Como es un prototipo, se asume que la pieza se puede mover libremente a izquierda y derecha, que no chocará con otra pieza ni completará una línea al hacerlo.
3. Indicarle a la pieza actual que baje lo más posible

Las **reglas** que el juego debe implementar en este prototipo son:

1. La pieza siempre empieza en la fila 1 (más arriba), en la mitad del tablero, redondeando a izquierda de ser necesario. En el ejemplo del tablero de más arriba se puede ver que el tetromino T empieza en 4@1.
2. Las piezas se posicionarán a partir del elemento de más arriba a la izquierda. Por ejemplo, para el tablero mostrado más arriba, la posición de esa pieza tetromino T es 4@1.
A partir de la posición de la pieza es que se conoce las posiciones del resto de los elementos de la misma. Para el tetromino T que empieza en 4@1, las posiciones serían: 4@1, 5@1, 6@1 y 5@2, que se obtiene creando una colección con: posición, posición + (1@0), posición + (2@0) y posición + (1@1)
3. La secuencia de figuras se decide de manera random, enviando el mensaje *#nextIntInteger*: a un objeto de tipo Random. Según el número obtenido se crea una pieza de la siguiente manera:
 - 1 → Tetromino O
 - 2 → Tetromino T
 - 3 → Tetromino I
 - 4 → Tetromino L
 - 5 → Tetromino Z
4. Las piezas no pueden salir del tablero.
5. Una pieza baja hasta la última fila (más abajo) o hasta que choca con otra pieza.
6. Cuando una pieza no puede bajar más, todas las posiciones de la pieza pasan a estar ocupadas.
7. Cuando una pieza no puede bajar más, hay que sacar todas las filas que estén completas con posiciones ocupadas en todas sus columnas.
8. Cuando una fila se saca por estar completa, todas las filas por arriba de ella bajan una fila.
9. Se suma 10 puntos al puntaje total, que empieza en 0, por cada línea completada.
10. El juego termina cuando la pieza actual no puede bajar de su posición de inicio.

Debido a que es un prototipo, no es necesario implementar lo siguiente:

- Rotación de piezas
- Sumar puntos por pieza que llega al final

Trabajo a realizar dependiendo de qué están recuperando:

- Para 1er Parcial: Además de representar el juego y las piezas, implementar acciones 1 y 2, e implementar las reglas 1 a 4 inclusive.
- Para 2do Parcial: Hacer todo.

- Para 1er y 2do Parcial: Hacer todo más:
Reemplazar la regla 4) por: El comportamiento de las piezas cuando chocan las paredes laterales varía de la siguiente manera:
 - Tetromino O y T: Aparecen del otro lado del tablero, misma fila.
 - Tetromino I y L: Rebotan 2 columnas. Por ejemplo, si una pieza Tetromino I choca con la pared izquierda, su nueva posición debería estar en la columna 3, misma fila en la que estaba.
 - Tetromino Z: Se va a la columna de la posición inicial manteniendo la fila.

Tips:

- Para obtener un número random se debe utilizar el mensaje `#nextIntInteger`: que está implementado en **Random**
- Recordar que la font default de Cuis no es proporcional, puede suceder que la representación string del tablero no se vea bien alineada. Si quieren usar fonts mono-espaciadas vayan al menú del desktop, elijan "Preferences > Set System Font ... > DejaVu > DejaVuSansMono" (o cualquier otra font Mono, de mono-espaciada). En algunos casos puede no quedar seleccionada esa font y solo la instalará, en ese caso se debe seleccionarla como fuente a usar, para ello hagan "Preferences > Set System Font ... > DejaVu Sans Mono" de la parte del menú que dice "Installed Fonts"

Trabajo a realizar

Implementar el prototipo del juego del **Tetris** pedido usando TDD y las heurísticas de diseño vistas en la materia, **sacando código repetido, reemplazando ifs por polimorfismo donde haga falta, etc**

Entrega:

1. Entregar el fileout de la categoría de clase **2024-2C-Recuperatorio** que debe incluir toda la solución (modelo y tests). El archivo de fileout se debe llamar: **2024-2C-Recuperatorio.st**
2. Entregar también el archivo que se llama **CuisUniversity-nnnn.user.changes**
3. Probar que el archivo generado en 1) se cargue correctamente en una imagen "limpia" (o sea, sin la solución que crearon. Usen otra instalación de CuisUniversity/imagen si es necesario) y que todo funcione correctamente. Esto es fundamental para que no haya problemas de que falten clases/métodos/objetos en la entrega.
4. Deben entregar usando el siguiente form: <https://tinyurl.com/inge1-recu-2c-2024>
5. Deberán obtener un ID de un docente, a completar en el form al momento de subir la info.
6. De forma alternativa si no pudiste entregar con el form, realizar la entrega enviando mail a: **entregas@isw2.com.ar** con el **Subject: LU nnn-aa - Solución Recuperatorio 2c2024.**

En caso de rebotar el envío, reintentar comprimiendo los adjuntos.

7. **RECOMENDACIÓN IMPORTANTE:** Salvar la imagen de manera frecuente o con el autosave
8. Se asume que a esta altura de la cursada saben trabajar con la imagen, recuperarla, recuperar código fuente, revertir cambios y demás incidencias que pudieran ocurrir durante el exámen.

Revisen bien los puntos de arriba. Cualquier error en los nombres o formato podrían ser penalizados en la nota.

IMPORTANTE: No retirarse sin tener el ok de los docentes de haber recibido la resolución por algún medio.

CERRAR EL TRABAJO A LAS 21:50.

LAS ENTREGAS RECIBIDAS DESPUÉS DE LAS 22:00 HRS NO SERÁN TENIDAS EN CUENTA