Feedback Tarea Sesión 1

1. Hay que **leer muy bien las instrucciones** del enunciado. Parece obvio pero no se hace y se olvidan cosas importantes, como renombrar el proyecto (dentro del proyecto).
2. Hay que renombrar el **proyecto** dentro del Eclipse.
    a. Botón derecho refactor/rename
    apellido1_apellido2_nombre_lab01_task_game2048
3. Para entregar hay que **exportar el proyecto** desde eclipse export/General/ArchiveFile botón Browse/ y colocarse en la carpeta donde vamos a exportar y to archiveFile **apellido1_apellido2_nombre_lab01**
4. Hay que eliminar siempre los warnings. En las pruebas daba el siguiente warning

```
7  import uo.mp.lab01.game.model.Game2048;
8
9
10 public class Game2048WithIntParamTest {
11     /*
12      * Pruebas del contructor con parametro el tamaño del tablero
13      * Casos de uso:
14      * 1-El parametro pasado al contructor es el minimo
15      * 2-El parametro pasado al contructor es el maximo
16      * 3-El parametro pasado al contructor es un valor entre 2 y 5
17      * 4-El parametro pasado al contructor es menor que el minimo
18      * 5-El parametro pasado al contructor es mayor que el maximo
19      */
20
21     /**
22      * Caso 1:El parametro pasado al contructor es el minimo
23      */
24     @Test
25     public void WithIntParamCase1() {
26         Game2048 game = new Game2048(Game2048.MIN_SIZE);
27         assertEquals(game.getBoard().length,Game2048.MIN_SIZE);
28         assertEquals(game.getBoard()[0].length,Game2048.MIN_SIZE);
29     }
```

   Es porque las pruebas están en el mismo paquete que Game2048 (aunque uno en src y otro en test). Al compilar, los junta en el mismo paquete, por lo que no necesita importarlo.

5. En las primeras líneas de código están los imports. Si son muchos no se ven y hay que pinchar en el signo más para verlos. Hay que eliminar todos los que están en amarillo porque no se están usando.
6. No se debe usar checkParam. Se usarán los métodos del proyecto Util
7. Cómo quitar el warning del siguiente código: Poniendo solo la llamada new Game2048(param)

```
@Test
public void boardParamNull() {
    int[][] param = null;
    try {
        Game2048 game = new Game2048(param);
    } catch (IllegalArgumentException e) {
        assertEquals(e.getMessage(), "Parametro incorrecto");
    }
}
```

8. Los métodos que se implementen sólo para usarse en las pruebas se definen con modificador de paquete (protected o incluso mejor simplemente sin modificador)

9. Siempre que se devuelva un array, se hace una copia defensiva. Se puede usar Arrays.copyOf. Si es arrayList se usa new ArrayList(lista). Hacen una copia superficial
10. Siempre que se reciba un array, se hace una copia defensiva.
11. Si en lugar de array de una dimensión es de dos, no sirve Arrays.copyOf ni clone porque no hacen copia de los vectores que componen las filas de la matriz. Hay que hacer un método que haga la copia de toda la estructura.
12. Se debe usar siempre StringBuilder para generar un String cuando hay que realizar muchas concatenaciones (por ejemplo, cuando generamos un string de una matriz). La concatenación de Strings provoca la creación de uno nuevo. StringBuilder con el append amplía el tamaño sin crear uno nuevo. Es por tanto mucho más eficiente. (%-5s) significa espacio de 5 caracteres justificado a la izquierda para un String.

```
public String toString() {
    StringBuilder sb = new StringBuilder();
    for (int row = 0; row < this.board.length; row++) {
        for (int col = 0; col < this.board.length; col++) {
            sb.append(String.format("%-5s", board[row][col]));
        }
        sb.append("\n");
    }
    return sb.toString();
}
```