Universidad de Alcalá

Escuela Politécnica Superior

Universidad de Alcalá

PRÁCTICA 2 – 16384 Scala

Ampliación de programación avanzada

Grado en Ingeniería Informática – Curso 2018/2019

Eduardo Graván Serrano – 03212337L

Marcos Barranquero Fernández – 51129104N

INTRODUCCIÓN

La práctica consiste en desarrollar un juego similar al 2048 usando el lenguaje de programación Scala de una forma lo más funcional posible. No se permite el uso de vars, objetos mutables ni métodos con efectos colaterales.

Se entregan dos archivos de código:

- El primero de ellos tiene implementado colores en la terminal y debería ser considerado como el principal.
- El segundo de ellos es exactamente el mismo proyecto, pero sin los colores implementados, y debería ser usado solo si el primero da problemas.

Se hace esta separación en dos archivos ya que dentro de la IDE que hemos utilizado, IntelliJ, los colores funcionan correctamente, pero fuera de ella (se ha probado Eclipse y CMD de Windows) los colores no funcionan e imprimen basura que dificulta la lectura del tablero. Pedimos que se considere el primer archivo de código como el principal y, sólo en el caso de que los colores no funcionen en la terminal en la que se corra el código, se use el segundo.

QUÉ SE ENTREGA

A parte de esta memoria descriptiva, se entregan los dos archivos de código descritos anteriormente.

En cuanto a las funcionalidades básicas que se piden en el enunciado, se han desarrollado todas ellas excepto el movimiento del tablero usando las flechas de movimiento del programa, ya que para poder implementar esto se necesitarían lanzar ventanas de una GUI o el uso de librerías externas que modificasen la terminal poniéndola en modo raw y que fuese capaz de leer esas teclas como input.

Se ha desarrollado también una optimización sobre el método de elección de mejor jugada sobre el modo de juego automático. Esta optimización consiste en mirar la mejor puntuación que podría conseguir la máquina con sus dos siguientes jugadas. Esto es, si consigue puntos con el primer movimiento, los suma con los puntos que podría conseguir con los posibles movimientos del siguiente paso y saca el máximo entre todos estos movimientos. Una vez hecho esto, saca el máximo de puntos posibles entre todas estas situaciones y elige en base a esto.

No se ha hecho implementación con librerías gráficas.

EJECUCIÓN SECUENCIAL

Si se ejecuta el programa, la secuencia de ejecución es la siguiente:

1. Se da la bienvenida al programa y se pide el nivel de dificultad con el que se quiere jugar. Si se da un valor no válido de dificultad, se sigue preguntando hasta que se dé un valor válido (entre 1 y 4).
2. Se genera una matriz aleatoria con las dimensiones adecuadas según la dificultad escogida.
3. Se pregunta por qué modo de ejecución se quiere usar. Esto es, se debe elegir entre modo manual o automático. Se repite la pregunta hasta que se dé un valor válido (a = automático; m = manual).
4. En el caso del bucle de juego manual: Se entra en el bucle de juego. En el podemos ver el numero de movimientos que hemos hecho hasta ahora, la puntuación que tenemos por ahora, y el número de colisiones que se han producido hasta el momento. También podemos ver el tablero en este instante de tiempo y se nos pide un input (w = arriba, s = abajo, a = izquierda, d = derecha). Mientras el input no sea correcto, se sigue preguntando. Una vez se ha elegido una dirección, se

ejecuta el código correspondiente a ese movimiento, que se compone de la búsqueda de semillas con mismo valor, su suma y el desplazamiento en el sentido que se haya escogido. Si el movimiento no modifica el tablero, es decir, no produce ningún movimiento en sus casillas, se rechaza el movimiento y se pide otro distinto.

5. Se vuelve a llamar al punto 4, habiendo actualizado el número de movimientos, la puntuación, y las colisiones.
6. Si el tablero se llena y se detecta que no hay movimientos posibles, salta un mensaje que indica la derrota y se sale del bucle de juego, mostrando la puntuación y sustrayendo una vida.
7. Si se detecta el valor objetivo en el tablero (16384), se muestra un mensaje de victoria, se muestra la puntuación, se añade una vida por haber ganado y se vuelve a empezar si el usuario quiere seguir jugando.
8. Este bucle de juego se ejecuta hasta que el usuario haya completado 3 tableros, se quede sin vidas, o decida salir del programa introduciendo el carácter 'x'.
9. En el caso de que se escoja el modo automático: el funcionamiento es similar, pero en vez de elegir la dirección manualmente, la máquina ejecuta su función de calculo de mejor movimiento y elige por sí misma. Para avanzar en la ejecución hay que pulsar Enter. En caso de que se quiera salir, se puede introducir el carácter 'x'.

CÓMO JUGAR

Al iniciar el juego, se debe introducir la dificultad del juego: 1,2,3 o 4.

Después, pregunta si se quiere modo manual o automático (m/a).

Tras esto, comienza el juego y se muestra el tablero. Si se eligió modo manual, se debe mover el tablero escribiendo W,A,S,D y pulsando la tecla Enter. Si se eligió el modo automático, las jugadas se eligen solas, por lo que sólo se debe pulsar Enter. Si se gana o pierde, se pregunta si se quiere jugar otra vez. Se debe responder escribiendo s o n, de si o no.

EXPLICACIÓN MODULAR

El programa podría dividirse en los siguientes módulos:

FUNCIONES DE CONFIGURACIÓN DEL MODO DE JUEGO

En el inicio del archivo de código hay variables que dependen de la entrada por teclado que se haga. Para establecer la dificultad, se hace una llamada a una función que da la bienvenida al programa y pregunta por la dificultad con la que queremos jugar. Esta dificultad se almacena en una val global, que será usada en una serie de métodos para determinar el número y tipo de semillas que se van a usar durante la ejecución.

También se hace una llamada a una función que se encarga de preguntar que modo de juego queremos ejecutar, entre manual o automático. Dependiendo de la respuesta se llamará a la función correspondiente. En total, se establecen:

- Dimensión de la matriz
- Lista de semillas posibles
- Número de semillas iniciales

FUNCIONES AUXILIARES DE TRATAMIENTO DE LISTAS

Tenemos una serie de funciones pequeñas que sustituyen a los métodos nativos de listas. Estas funciones se encuentran al comienzo del documento, y son utilizadas auxiliarmente por otros métodos. Entre otras, destacan:

- Strip: dada una lista, la devuelve quitando todos los 0s.
- Revertir: Dada una lista, la invierte.
- listaDeCeros: devuelve una lista con el nº de ceros especificados en la llamada.
- Recortar: Dada una lista, devuelve una sublista que comprende las posiciones lista[desde:hasta], ambas incluidas.
- Longitud: devuelve la cantidad de elementos en la lista dada.
- Eliminar: elimina la primera aparición de un elemento en la lista.
- Disyunción: devuelve los elementos no comunes de dos listas.
- getElemento: dada una lista y un índice, devuelve el elemento en esa posición en la lista.
- setElemento: dada una lista y un índice, escribe el elemento en esa posición en la lista.
- Contiene: dada una lista y un elemento, devuelve true si el elemento está en la lista.
- sonIguales: dadas dos listas, devuelve true si son iguales.
- Máximo: dada una lista, devuelve el entero con mayor valor que se encuentre en esta.

Todas estas funciones funcionan de manera recursiva y comprueban inputs en busca de errores. Son seguras y no producen efectos colaterales.

FUNCIONES DE TABLERO

Contamos con las siguientes funciones auxiliares para el tratamiento de tablero:

GETTERS, SETTERS E INICIALIZACIÓN

- GetFila y GetColuma: dado un tablero, devuelven una lista con la fila o columna especificada como argumento.
- SetFila y SetColumna: dado un tablero, una fila y una nueva fila o columna, escriben esa nueva fila o columna especificada en la fila o columna indicada.
- Rellenar tablero vacío: dado un tablero vacío, escribe las seeds iniciales en este, escribiendo en posiciones aleatorias las seeds posibles de la lista de seeds.
- Crear Tablero: dada una dificultad, devuelve un tablero con la dimensión apropiada y con las seeds iniciales.
- Add Semillas: añade el nº de semillas especificadas al tablero, tomándolo de la lista de posibles semillas.

COMPROBADORES

- EstaLleno: devuelve true si el tablero está lleno, es decir, no contiene ningún 0.
- Movimientos posibles: dado un tablero, comprueba si hay movimientos posibles en el tablero, es decir, si las casillas pueden combinarse con sus adyacentes. Para que funcione correctamente, solo se puede llamar a este método si se ha comprobado que el tablero este lleno previamente.
- Seguir Jugando: devuelve true si se puede seguir jugando, es decir, si el tablero no está lleno o lo está, pero se pueden realizar movimientos.
- SiONo y modoJuego piden introducir caracteres por teclado, por lo que se ha optado separarlos a una función aparte para poder comprobar la entrada y en caso de error realizar una llamada recursiva.

OPERACIONES

- sumarDerechaAux: realiza la operación de sumar las seeds posibles.
- desplazarDerechaAux: realiza la operación de desplazar las seeds tras haber sido sumadas.
- operarFila: realiza la operación de suma y desplazamiento de una fila.
- calculaColisiones: dado el tablero de antes y el tablero de después de ejecutar el movimiento, devuelve el nº de colisiones que ha habido tras mover, es decir, la diferencia de 0s que hay.
- Mover Dirección: tenemos una función para mover en cualquier dirección. Esencialmente consiste en, dado un tablero, para cada fila o columna, tomadas con getFila o getColumna, realizar la operación de sumar y desplazar, y llamar recursivamente con el tablero modificado hasta que se hallan operado todas las filas o columnas. Para realizar desplazamientos a la izquierda o columnas hacia arriba, utilizamos un reverse para realizar las operaciones al revés y que cuadre el movimiento con el resto del tablero.
- calcularPuntuación: dada una fila, calcula la puntuación de esta antes de ser operada.
- calcularPuntuación por movimiento: dado un tablero y un movimiento (izquierda, derecha, etc), calcula la puntuación resultante de realizar dicho movimiento.
- Calcular mejor jugada: haciendo uso de funciones auxiliares, esencialmente calcula la puntuación máxima posible para cada movimiento. Si la puntuación de ese movimiento es distinta de 0, calcula la mejor puntuación posible de los posibles movimientos desde ese movimiento. Es decir, calcula la mejor puntuación posible tras esos dos movimientos.
- Pintar tablero: dibuja el tablero dado por pantalla.

JUEGO MANUAL

El juego se divide en dos funciones principales:

BUCLE JUEGO MANUAL

Recibe un tablero con las seeds iniciales, y contadores de movimientos, puntuación y colisiones. A partir de ahí seguirá el bucle descrito avanzando por el juego, hasta que devuelva true si se llega al objetivo o false si se pierde.

El bucle de ejecución consiste en lo siguiente:

1. Se muestra por pantalla el nº de movimientos, puntuación y colisiones.
2. Se verifica si se ha llegado al objetivo. De ser así, se devuelve true.
3. Se verifica si se puede seguir jugando. Si no se puede, devuelve false.
4. Si no se ha entrado en los pasos 2 y 3, se pide al usuario que pulse una tecla (W,A,S,D) para mover el tablero.
5. En función de la tecla pulsada, se verifica si el movimiento es válido.
    a. Si lo es, se llama recursivamente a bucle de juego manual pasando el tablero movido y los contadores actualizados.
    b. Si no lo es, se indica por pantalla y se llama recursivamente con los mismos argumentos.
6. Si se pulsa x, se sale del juego. Si se pulsa una tecla no válida, se llama recursivamente con los mismos argumentos.

JUGANDO CON VIDAS MANUAL

Esta función es la encargada de llevar el contador de vidas e inicializar los tableros, así como mostrar por pantalla mensajes de victoria, derrota, contar cuantas partidas se llevan ganadas, etc. También es la función que llama a bucle juego manual.

Mientras no se hayan ganado más de 3 veces seguidas, se sigue llamando a bucle juego manual y recogiendo el booleano que devuelve. En función de este mostrará por pantalla victoria o derrota y actualizará el contador de veces que se ha ganado y de vidas.

JUEGO AUTOMÁTICO

El bucle de juego automático es relativamente similar al de juego manual, pero con los siguientes cambios:

BUCLE JUEGO AUTOMÁTICO

Ahora, en lugar de leer el input del usuario, se llama a la función que calcula la mejor jugada posible a realizar, de forma que no debemos comprobar inputs. Para avanzar se puede mantener pulsado enter y si se quiere salir basta con pulsar x.

JUGANDO CON VIDAS AUTOMÁTICO

También es muy similar a la versión manual: en función del output de bucle juego manual se muestran mensajes de ganar o perder y se actualizan contadores de vidas y veces ganado. El usuario puede elegir si se quiere volver a jugar o salir.