SortedMap y TreeMap

SortedMap

Esta interfaz es muy parecida la interfaz Map. Tan solo se diferencia en que SortedMap permite que los elementos dentro del conjunto de la colección estén ordenados totalmente, facilitando por tanto su acceso en búsquedas y haciendo más rápida su consulta.

Así pues los elementos del sortedmap (mapa ordenado) están ordenados por sus elementos claves.

¿Cuándo usar la interface Map y cuándo usar la interface SortedMap?

Básicamente cuando vayamos a crear nuestra estructura de datos deberemos pensar en si necesitamos o es ventajoso mantener los datos ordenados de alguna manera para hacer más rápido el acceso. En este caso nos decantaremos por el uso de la interface SortedMap y las clases que la implementan.

Si no resulta ventajoso el mantener los datos ordenados o se trata de mapas donde no hay una gran cantidad de datos, la ordenación puede suponer un consumo de recursos innecesario y podemos optar por usar la interface Map y las clases que la implementan.

TreeMap

TreeMap es la clase que vamos a utilizar como implementación de la interface SortedMap. TreeMap se basa en una implementación en árbol que permite tener un mapa ordenado implementando así por tanto la interfaz TreeMap.

Para explicar mejor SortedMap y TreeMap lo haremos con un ejemplo en eclipse.

En este ejemplo tenemos 3 clases ; Jugador, Equipo, gestión de Equipo.

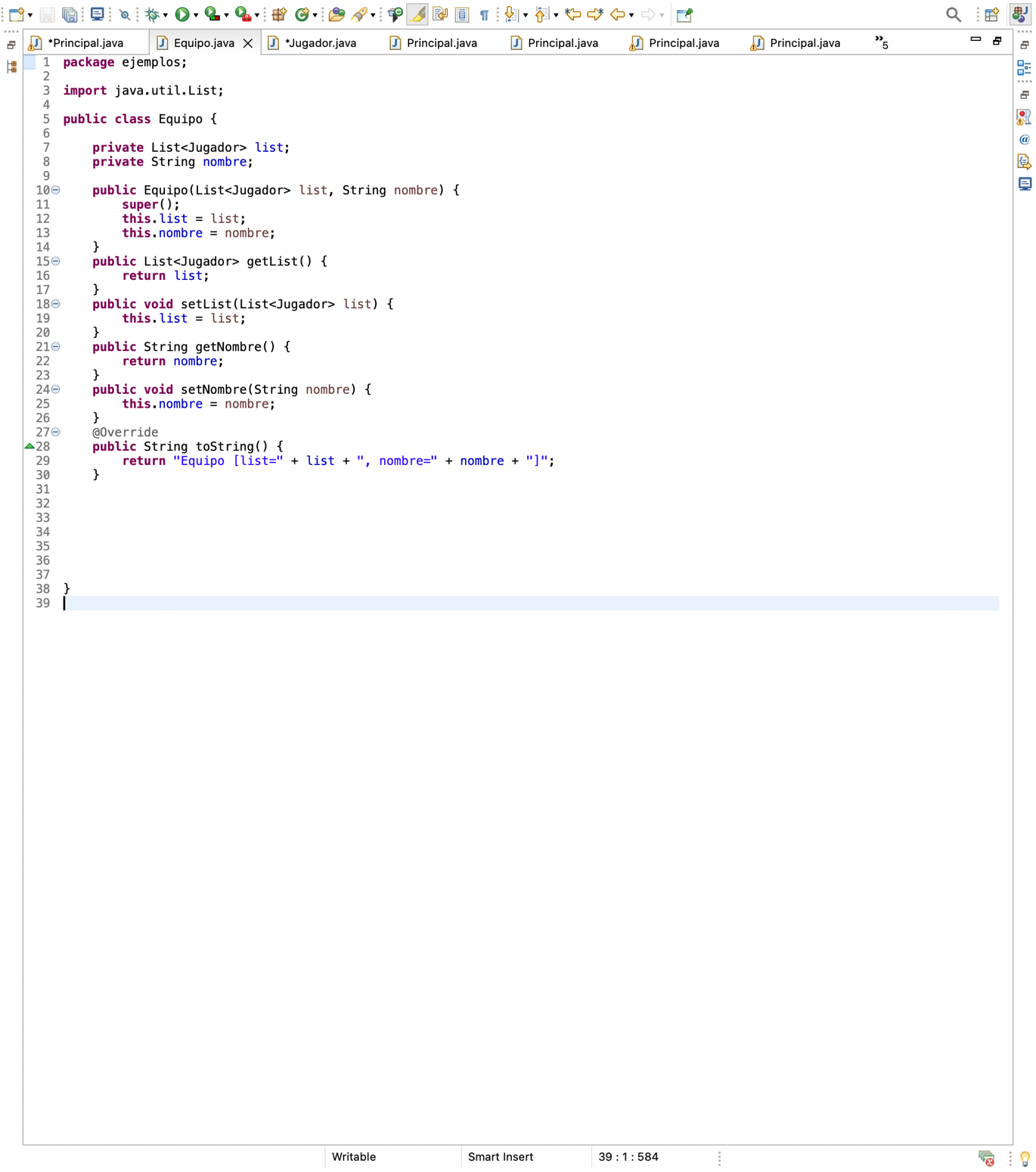
Nos situamos en la clase Jugador:



En la clase Jugador tenemos como atributos nombre, edad,posición…

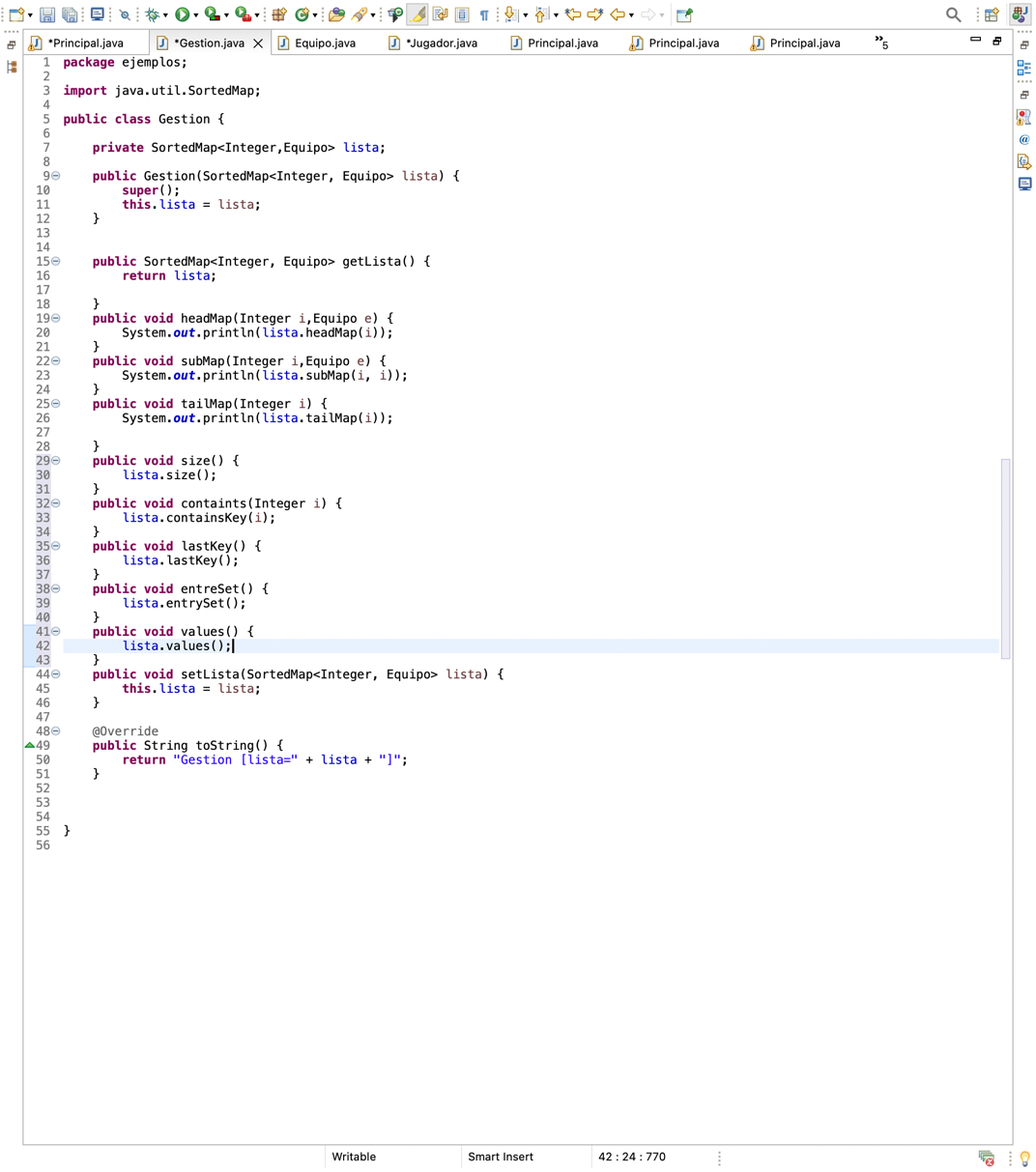
Y en esta clase implementamos la interfaz comparable

Esta interfaz nos va a permitir ordenarlos mas adelante .

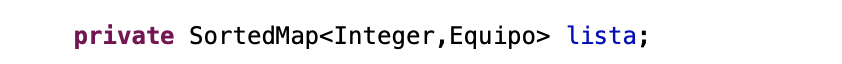
Nos situamos ahora en la clase Equipo:

En esta clase lo único importante a mencionar es que tenemos un atributo lista de jugadores , es decir , esto significa que un equipo esta compuesto por una lista de jugadores.

Y en clase Gestión tenemos:



En esta clase tenemos como atributo un SortedMap como clave Integer y como valor Equipo .



En esta clase tenemos los métodos que utilizaremos .

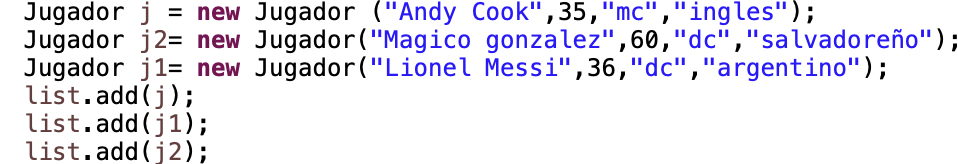
En la clase principal es donde veremos que utilidad tienen los métodos



Lo que hay en esta imagen he instanciado la interfaz SortedMap y TreeMap

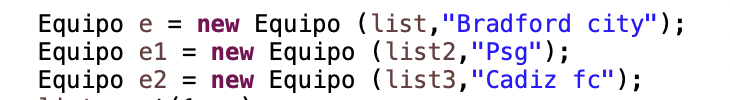


Después de hacer esto he instanciado un ArrayList 3 veces para guardar jugadores en tres equipos diferentes



Esto quiere decir que Andy ,mágico y Lionel pertenecerán a un equipo

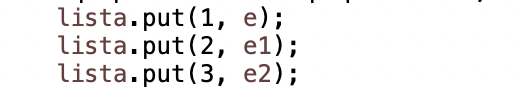
Una vez añadido los jugadores tenemos que añadir los jugadores a los equipos esto se hace de la siguiente forma .



En cada “list” hay Jugadores y lo que acabamos de hacer en añadirlos en los correspondientes equipos .

La función que tiene list es añadir jugadores al equipo.

Una vez añadido los jugadores a los equipos tenemos que añadirlos al mapa

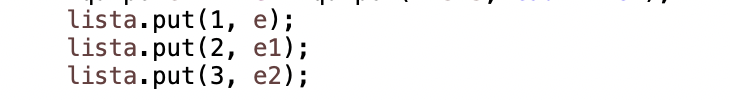


En esta imagen el 1,2,3 nos indican el puesto que ocupara en el map ,como por asi llamarlos un “id” .

Con esto ya estaría añadidos al map. Una vez añadido ya podemos utilizar los métodos que implementan estas interfaces.

METODOS

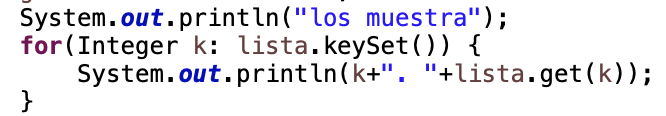
-El primer método lo acabamos de ver ,

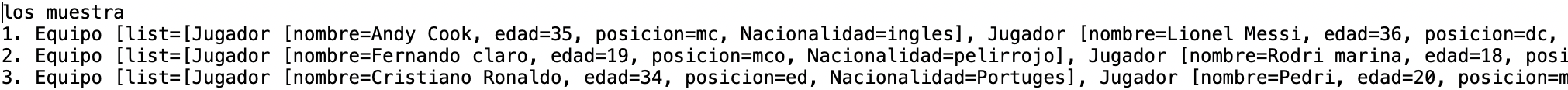


Este método lo implementa estas interfaces, que lo su función como ya hemos dicho antes es añadir Jugadores y Equipos en el mapa

Esto no es un método pero asi podemos mostrar a los jugadores y equipos por consola

Simplemente se utiliza para recorrer todas las claves de un mapa y asignar cada clave a una variable en cada recorrido

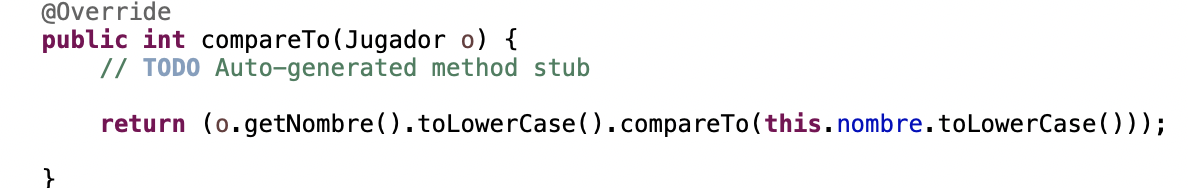


Este es el resultado:

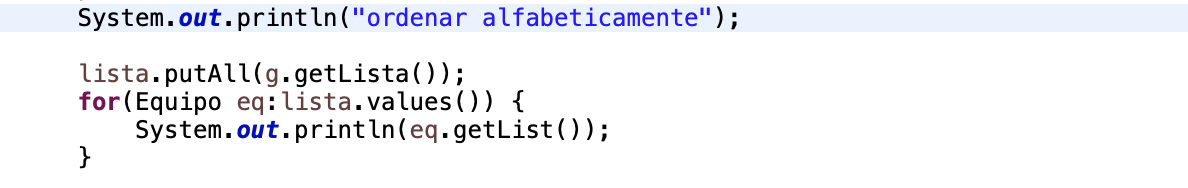
En la imagen no salen todos porque es una lista muy grande y no cabe en la imagen

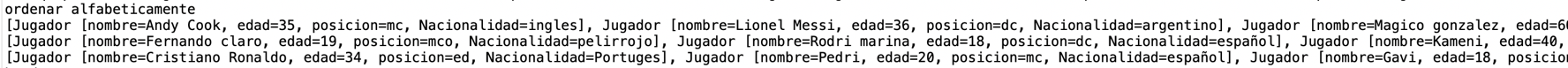
-Tenemos una forma de ordenar el map que es con la interfaz Comparable que es la que implementamos en la clase Jugador.

Una vez implementada esta interfaz nos obliga añadir un método que se llama compareTo que es el siguiente:

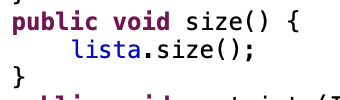


Basicamente lo que hace este método es comparar el nombre de jugadores ya que nosotros hemos elegido ordenarlo por nombres de jugadores .

Una vez terminado este método, para aplicarlo en el main se podría hacer asi :

El resultado es el siguiente :

-El método size :

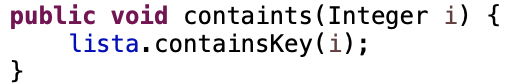


Este método lo que hace es que imprime por pantalla la cantidad de claves del mapa

En este caso tenemos 3 claves entonces en la pantalla nos tendría que salir el numero 3



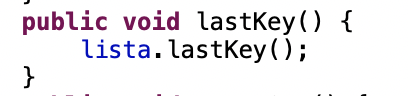
-El método containts:



Devuelve un true o false si el índices en el map en este caso nos tendría que devolver un true :



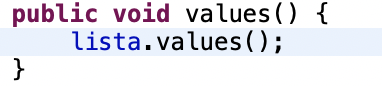
-El método lastKey:



Nos devuelve por pantalla la ultima clave del mapa. En este caso nos tendría que salir 3 porque 3 es el ultimo numero de la clave :



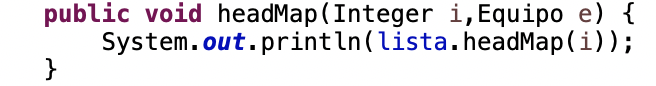
-El método values :



Este método nos devuelve por pantalla todos los valores del mapa :



-El método headMap:



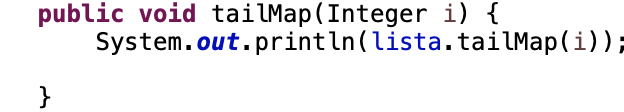
Lo que hace este método es que el valor que le des en la variable i (en el id del mapa)

Te mostrará los valores del mapa hasta llegar al valor del id => i.

En este caso solo quiero que llegue hasta el id 2 (exclusive):



-El método TailMap:



Este método hace algo parecido al método anterior , pero en vez de mostrarte hasta un índice del mapa este método te mostrar los valores desde el id que le hayas introducido hasta el ultimo.

En este ejemplo he puesto que muestre a partir del id 2 :

