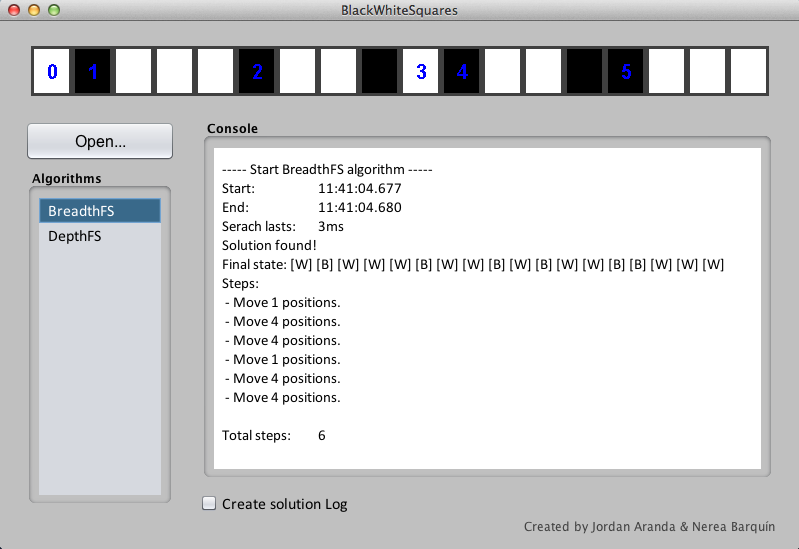
**Homework 3/4**

Question 3 of the general problema case description.

**BreadthFS**

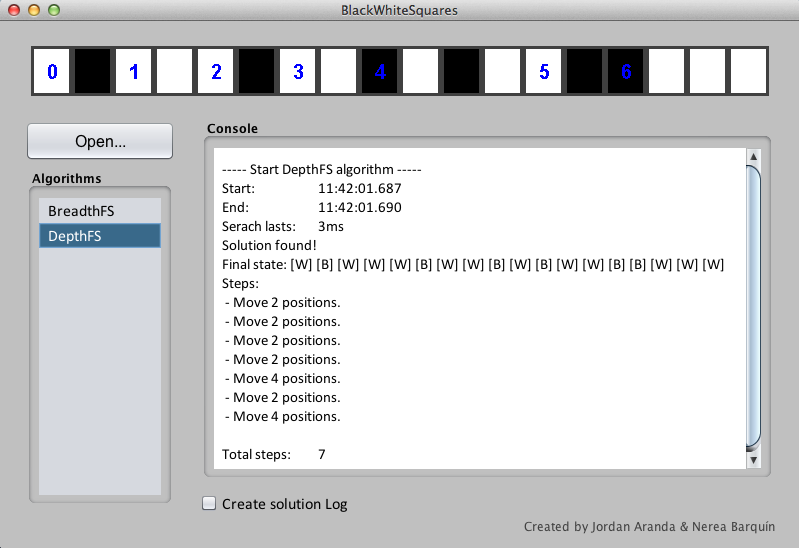


Branching factor “b” (Número máximo de sucesores) 🡪 2 nodos sucesores.

Maximum depth “m” 🡪 19

Shortest depth “d” 🡪 7

**DepthFS**



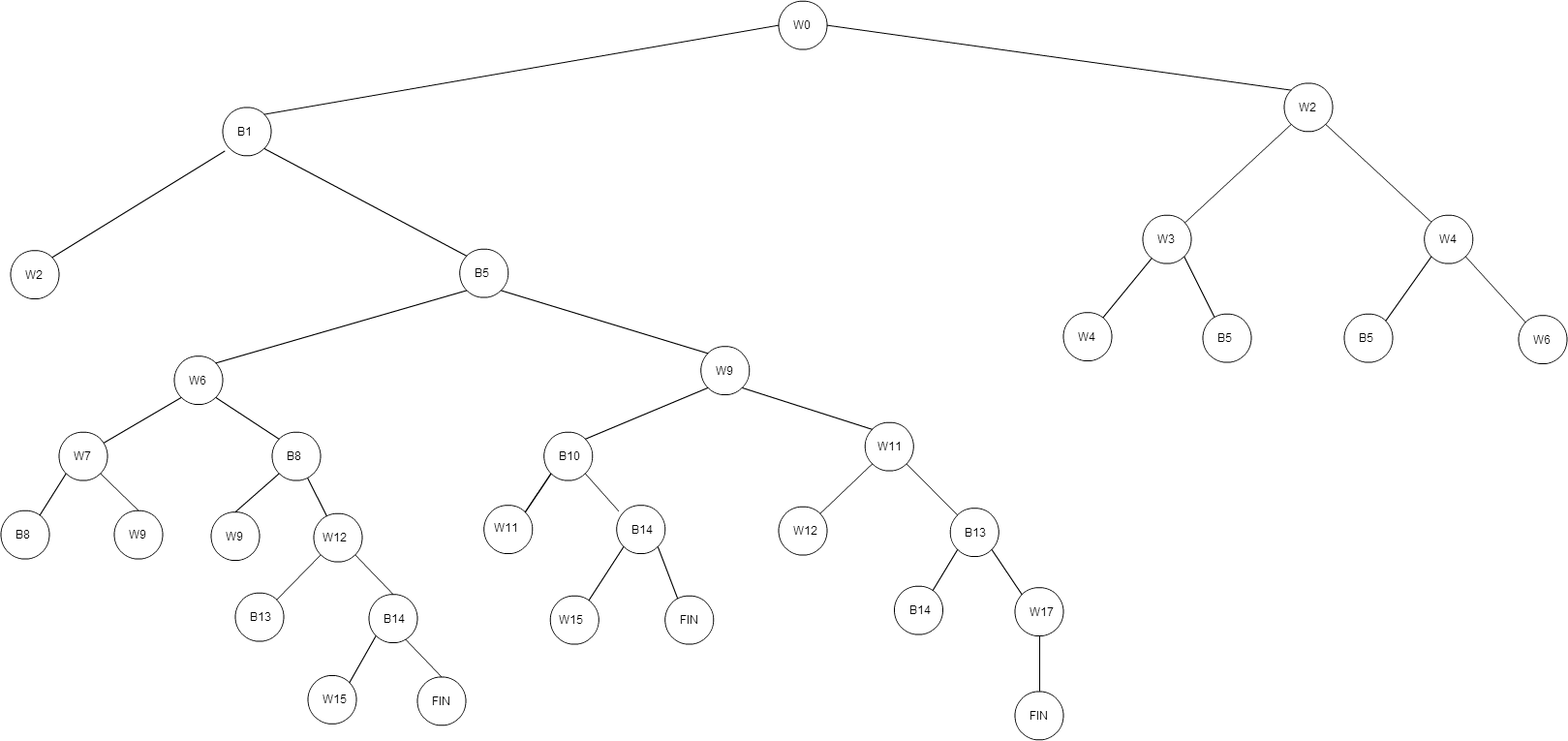
Branching factor “b” 🡪 2 nodos sucesores.

Maximum depth “m” 🡪 19

Shortest depth “d” 🡪 8

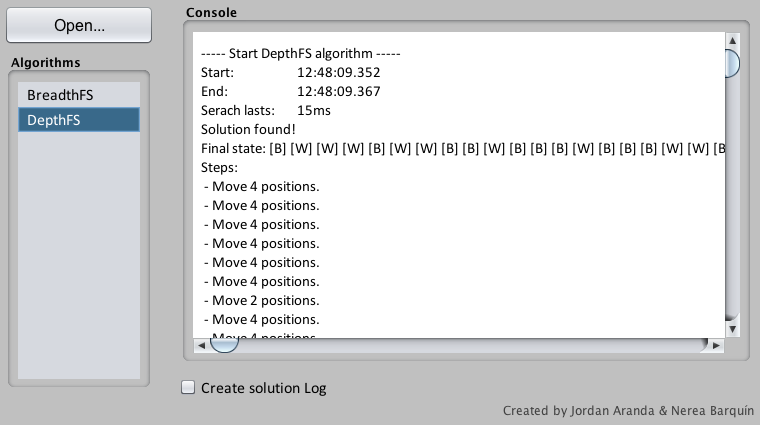
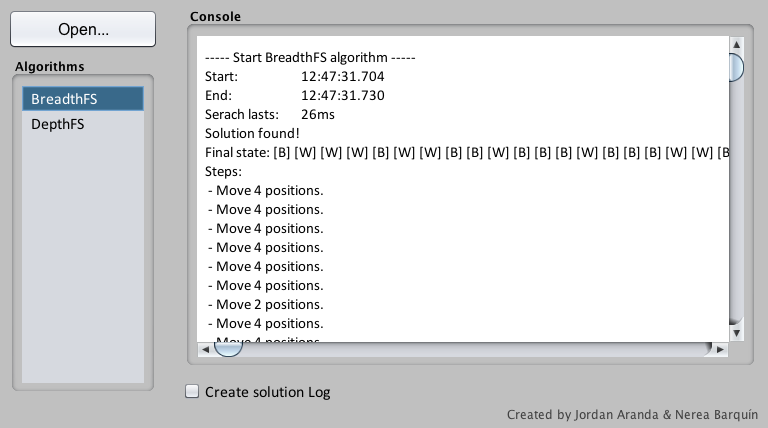
En este caso, el algoritmo óptimo es el de BreadthFS porque siempre que haya varias posibles soluciones devuelve la mejor, es decir, el menor número de acciones.

El algoritmo DepthFS no es completo porque no tiene en cuenta los nodos de un mismo nivel, sin embargo el algoritmo BreadthFS es completo porque va haciendo un barrido completo de los nodos del árbol.



Diferencias entre los algoritmos BreadthFS y DepthFS.

* El algoritmo BreadthFS inserta los operadores al final de la lista y van saliendo por el principio de esta (FIFO). Mientras que el algoritmo DepthFS inserta los operadores al principio y van saliendo por el final (LIFO). En ambos casos, lo prioritario se deberá insertar antes en la lista.
* En el caso del algoritmo BreadthFS encuentra el estado final con un coste de solución menor que el de DepthFS. El algoritmo DepthFS no tiene por que ser óptimo. Por otro lado, el algoritmo DepthFS es completo porque no tiene en cuenta los nodos de un mismo nivel.
* El algoritmo DepthFS utiliza menos tiempo y memoria que el BreadthFS para encontrar la solución del problema. Este hecho lo hemos podido comprobar realizando una prueba con un tablero de 1000 posiciones.



Preguntas pendientes:

* Como se puede mejorar el tiempo?
* Lista de operadores
* Insertar árbol de nodos