

```
10 public static boolean canWin(int[] arr, int leap) {
11     boolean answer = false;
12     int stack = 0;
13     boolean result = move(0, -1, arr, leap, answer, stack);
14     return result;
15 }
16
17 public static boolean move(int currIndex, int prevIndex, int[] arr, int leap, boolean answer, int stack) {
18     // base case START-----
19     if ((currIndex + leap) >= arr.length) {
20         System.out.println("yes possible");
21         answer = true;
22         return true;
23     }
24     if ((currIndex + 1) >= arr.length) {
25         answer = true;
26         return true;
27     }
28     // base case END-----
29
30     // step backwards
31     if ((currIndex - 1) >= 0) {
32         if ((arr[currIndex - 1] == 0) && (currIndex - 1) != prevIndex && (currIndex - 1) > stack) {
33             System.out.println("step backwards once: curr: " + currIndex + " prev: " + prevIndex);
34             move((currIndex - 1), currIndex, arr, leap, answer, stack);
35         }
36     }
37
38     // leap backwards
39     if ((currIndex - leap) >= 0) {
40         if ((arr[currIndex - leap] == 0) && (currIndex - leap) != prevIndex && (currIndex - leap) > stack) {
41             System.out.println("step backwards LEAP: curr: " + currIndex + " prev: " + prevIndex);
42             move((currIndex - leap), currIndex, arr, leap, answer, stack);
43         }
44     }
45
46     // Update Stack
47     // updating it here ensures we have traversed backwards as much as possible
48     stack = currIndex;
49     System.out.println("Stack updation reached: stackVal: " + stack);
50
51     // step forward
52     if ((currIndex + 1) < arr.length) {
53         if ((arr[currIndex + 1] == 0) && (currIndex + 1) != prevIndex && (currIndex + 1) > stack) {
54             System.out.println("step forward once: curr: " + currIndex + " prev: " + prevIndex);
55             move((currIndex + 1), currIndex, arr, leap, answer, stack);
56         }
57     }
58
59     // leap forward
60     if ((currIndex + leap) < arr.length) {
61         if ((arr[currIndex + leap] == 0) && (currIndex + leap) != prevIndex && (currIndex + leap) > stack) {
62             System.out.println("step forward LEAP: curr: " + currIndex + " prev: " + prevIndex);
63             move((currIndex + leap), currIndex, arr, leap, answer, stack);
64         }
65     }
66     return answer;
```

```
67 | }
```