```java
import java.util.Arrays;

public class MakeMove {

    public static int[][]currentBoard = new int[4][4];
    public static int numMoves;

    public static void addingTiles(int[][] board, String dir) {
        int[][] tempArray = board;
        switch (dir) {
            case "a":
                //movedArray temporarily stores the current board for comparison, if the two arrays
                //are equal, it means no movement was made, therefore it was an invalid move
                copyBoard();
                //Checks if numbers are the same, if they are then you add them
                for (int i = 0; i < 4; i++) {
                    for (int j = 0; j < 3; j++) {
                        if (tempArray[i][j] == tempArray[i][j + 1]) {
                            tempArray[i][j] += tempArray[i][j + 1];
                            tempArray[i][j + 1] = 0;
                        }
                    }
                }
                //Loops through array to check if there is a zero to the left of the element,
                //If there is, you swap them, essentially shifting to the left
                for (int i = 0; i < tempArray.length; i++) {
                    for (int j = 0; j < tempArray[i].length; j++) {
                        moveLeft(board);
                    }
                }

                if(!Arrays.deepEquals(currentBoard, tempArray)) {
                    Board.updateBoard();
                    numMoves++;
                }
                break;

```

```
38              case "d":
39                  copyBoard();
40                  //Check if numbers are the same, if they
   are, add them
41                  for (int i = 0; i < 4; i++) {
42                      for (int j = 3; j > 0; j--) {
43                          if(tempArray[i][j] == tempArray[i][
   j - 1]){
44                              tempArray[i][j] += tempArray[i
   ][j-1];
45                              tempArray[i][j-1] = 0;
46                          }
47                      }
48                  }
49                  //Loops through array to check if there is
   a zero to the right of the element,
50                  //If there is, you swap them, essentially
   shifting to the right
51                  for(int i = 0; i < tempArray.length; i++){
52                      for(int j = 0; j < tempArray[i].length
   ; j++){
53                          moveRight(board);
54                      }
55                  }
56
57                  if(!Arrays.deepEquals(currentBoard,
   tempArray)) {
58                      Board.updateBoard();
59                      numMoves++;
60                  }
61
62                  break;
63              case "w":
64                  copyBoard();
65                  //Check if numbers in the row above are the
    same, if they are, add them
66                  for (int i = 1; i <= 3; i++) {
67                      for (int j = 0; j < 4; j++) {
68                          if(tempArray[i][j] == tempArray[i
    - 1][j]){
69                              tempArray[i-1][j] += tempArray[
   i][j];
70                              tempArray[i][j] = 0;
71                          }
72                      }
73                  }
74                  //Loops through array to check if there is
   a zero above the element,
```

```
 75                 //If there is, you swap them, essentially
    shifting "above"
 76                 for (int i = 0; i < tempArray.length; i
    ++) {
 77                     for (int j = 0; j < tempArray[i].
    length; j++) {
 78                         moveUp(board);
 79                     }
 80                 }
 81
 82                 if(!Arrays.deepEquals(currentBoard,
    tempArray)) {
 83                     Board.updateBoard();
 84                     numMoves++;
 85                 }
 86                 break;
 87             case "s":
 88                 copyBoard();
 89                 //down
 90                 for (int i = 3; i > 0; i--) {
 91                     for (int j = 0; j < 4; j++) {
 92                         if(tempArray[i][j] == tempArray[i
     - 1][j]){
 93                             tempArray[i][j] += tempArray[i
    -1][j];
 94                             tempArray[i-1][j] = 0;
 95                         }
 96                     }
 97                 }
 98                 //Loops through array to check if there is
     a zero below the element,
 99                 //If there is, you swap them, essentially
    shifting "down"
100                 for (int i = 0; i < tempArray.length; i
    ++) {
101                     for (int j = 0; j < tempArray[i].
    length; j++) {
102                         moveDown(board);
103                     }
104                 }
105                 if(!Arrays.deepEquals(currentBoard,
    tempArray)) {
106                     Board.updateBoard();
107                     numMoves++;
108                 }
109                 break;
110         }
111     }
```

```
112
113     //2D array copy method
114     public static int[][] copyBoard(){
115         for (int i = 0; i < 4; i++) {
116             for (int j = 0; j < 4; j++) {
117                 currentBoard[i][j] = Board.board[i][j];
118             }
119         }
120         return currentBoard;
121     }
122
123     //The methods below all shift elements in the array
    according to the desired direction
124     public static void moveLeft(int[][] array){
125         int[][] tempArray = array;
126         int tempValue;
127         for(int i = 0; i < tempArray.length; i++){
128             for(int j = 3; j >= 1; j--){
129                 if((tempArray[i][j] != 0 ) && (tempArray[i
    ][j-1] == 0)){
130                     tempValue = tempArray[i][j-1];
131                     tempArray[i][j-1] = tempArray[i][j];
132                     tempArray[i][j] = tempValue;
133                 }
134             }
135         }
136     }
137
138     public static void moveRight(int[][] array){
139         int[][] tempArray = array;
140         int tempValue;
141         for(int i = 0; i < tempArray.length; i++){
142             for(int j = 0 ; j < 3; j++){
143                 if((tempArray[i][j] != 0 && tempArray[i][j
    +1] == 0)){
144                     tempValue = tempArray[i][j+1];
145                     tempArray[i][j+1] = tempArray[i][j];
146                     tempArray[i][j] = tempValue;
147                 }
148             }
149         }
150     }
151
152     public static void moveUp(int[][] array){
153         int[][] tempArray = array;
154         int tempValue;
155         for(int i = 1; i <= 3 ; i++){
156             for(int j = 0; j < 4; j++){
```

```
157                 if(tempArray[i][j] != 0 && tempArray[i-1][
    j] == 0){
158                     tempValue = tempArray[i - 1][j];
159                     tempArray[i-1][j] = tempArray[i][j];
160                     tempArray[i][j] = tempValue;
161                 }
162             }
163         }
164     }
165 
166     public static void moveDown(int[][] array){
167         int[][] tempArray = array;
168         int tempValue;
169         for(int i = 2; i >= 0; i--){
170             for(int j = 0; j < 4; j ++){
171                 if(tempArray[i][j] != 0 && tempArray[i + 1
    ][j] == 0){
172                     tempValue = tempArray[i + 1][j];
173                     tempArray[i+1][j] = tempArray[i][j];
174                     tempArray[i][j] = tempValue;
175                 }
176             }
177         }
178     }
179 
180 
181 
182 
183 
184 }
185 
```