Macadine, Julius Caezar
2020–06459

CMSC 122 FINAL PROJECT

Algorithm Report

Data Structured Used: Array

For the Space Complexity:

Since the algorithm contains an array and it is the only data structure that is present in the algorithm, the space complexity of the algorithm is O(n). It is because the space complexity of the other initializations are O(1), and O(1) is insignificant against O(n) as n approaches infinity.

For the Time Complexity:

Algorithm Used:

Here is the general algorithm that I used in creating the Modified Sliding Puzzle Game:

Algorithm Modified Sliding Puzzle Game

Start

1 Set the moveCounter to 0. → **O(1)**

2 Set the image icons needed → **O(1)**

3 Set the necessary audios → **O(n)** *because the background music loops endlessly unless the user terminates the game*

4 Initialize the GUI → **O(1)**

5 Initialize the array (valueHolder[]) that will contain the value of each tile. → **O(1)**

6 Enter the game → **O(1)**

7 Shuffle the puzzle → **O(n)**

8 Check if the puzzle is solved (This is the time where the player switch the tile values to solve the puzzle and each move increases moveCounter by 1). → **O(n)** *since the player may play the game forever*

9 If the puzzle is solved, go to line 10 else go back to line 7 → **O(1)**

10 Declare the player as winner and display the total number of moves that they have taken to solve the puzzle → **O(1)**

End

If we add the total operations of the algorithm, we can come up with O(1 + 1 + n + 1 + 1 + n + n + 1 + 1) = O(6 + 3n). Since 6 and the coefficient 3 of n is insignificant against n as n approaches infinity, we can say that the time complexity of the algorithm is O(n).

Here is the algorithm for switching the tile values:

Algorithm Switching Tile Values ()

Start

1 If the player clicks the first tile: → **O(1+1)** → **O(1)**
2 set temp = the valueHolder[0] → **O(1)**
3 if valueHolder[1] == 9 → **O(1+1+1+1)**
4 valueHolder[0] = valueHolder[1] → **O(1)**
5 valueHolder[1] = temp → **O(1)**
6 moveCounter = moveCounter + 1 → **O(1)**
7 update the images of the tiles based on valueHolder[]→ **O(1)**
8 else if valueHolder[3] == 9 → **O(1+1+1+1)** → **O(1)**
9 valueHolder[0] = valueHolder[3] → **O(1)**
10 valueHolder[3] = temp → **O(1)**
11 moveCounter = moveCounter + 1 → **O(1)**
12 update the images of the tiles based on valueHolder[] → **O(1)**
13 If the player clicks the second tile:→ **O(1+1)** → **O(1)**
14 set temp = the valueHolder[1] → **O(1)**
15 if valueHolder[0] == 9:→ **O(1+1+1+1)** → **O(1)**
16 valueHolder[1] = valueHolder[0]→ **O(1)**
17 valueHolder[2] = temp→ **O(1)**
18 moveCounter = moveCounter + 1→ **O(1)**
19 update the images of the tiles based on valueHolder[]→ **O(1)**
20 else if valueHolder[2] == 9:→ **O(1+1+1+1)** → **O(1)**
21 valueHolder[1] = valueHolder[2]→ **O(1)**
22 valueHolder[2] = temp→ **O(1)**
23 moveCounter = moveCounter + 1→ **O(1)**
24 update the images of the tiles based on valueHolder[]→ **O(1)**
25 else if valueHolder[4] == 9:→ **O(1+1+1+1)** → **O(1)**
26 valueHolder[1] = valueHolder[4]→ **O(1)**

```
27                      valueHolder[4] = temp→ O(1)
28                      moveCounter = moveCounter + 1→ O(1)
29                      update the images of the tiles based on valueHolder[]→ O(1)
30      If the player clicks the third tile:→ O(1+1) → O(1)
31              set temp = the valueHolder[2]→ O(1)
32              if  valueHolder[1] == 9:→ O(1+1+1+1) → O(1)
33                      valueHolder[2] = valueHolder[1]→ O(1)
34                      valueHolder[1] = temp→ O(1)
35                      moveCounter = moveCounter + 1→ O(1)
36                      update the images of the tiles based on valueHolder[]→ O(1)
37              else if  valueHolder[5] == 9:→ O(1+1+1+1) → O(1)
38                      valueHolder[2] = valueHolder[5]→ O(1)
39                      valueHolder[5] = temp→ O(1)
40                      moveCounter = moveCounter + 1→ O(1)
41                      update the images of the tiles based on valueHolder[]→ O(1)
42      If the player clicks the fourth tile:→ O(1+1) → O(1)
43              set temp = the valueHolder[3]→ O(1)
44              if  valueHolder[0] == 9:→ O(1+1+1+1) → O(1)
45                      valueHolder[3] = valueHolder[0]→ O(1)
46                      valueHolder[0] = temp→ O(1)
47                      moveCounter = moveCounter + 1→ O(1)
48                      update the images of the tiles based on valueHolder[]→ O(1)
49              else if  valueHolder[4] == 9:→ O(1+1+1+1) → O(1)
50                      valueHolder[3] = valueHolder[4]→ O(1)
51                      valueHolder[4] = temp→ O(1)
52                      moveCounter = moveCounter + 1→ O(1)
53                      update the images of the tiles based on valueHolder[]→ O(1)
54              else if  valueHolder[6] == 9:→ O(1+1+1+1) → O(1)
55                      valueHolder[3] = valueHolder[6]→ O(1)
56                      valueHolder[6] = temp→ O(1)
57                      moveCounter = moveCounter + 1→ O(1)
58                      update the images of the tiles based on valueHolder[]→ O(1)
59      If the player clicks the fifth tile:→ O(1+1) → O(1)
60              set temp = the valueHolder[4]→ O(1)
```

```
61              if  valueHolder[3] == 9:→ O(1+1+1+1) → O(1)
62                      valueHolder[4] = valueHolder[3]→ O(1)
63                      valueHolder[3] = temp→ O(1)
64                      moveCounter = moveCounter + 1→ O(1)
65                      update the images of the tiles based on valueHolder[]→ O(1)
66              else if  valueHolder[1] == 9:→ O(1+1+1+1) → O(1)
67                      valueHolder[4] = valueHolder[1]→ O(1)
68                      valueHolder[1] = temp→ O(1)
69                      moveCounter = moveCounter + 1→ O(1)
70                      update the images of the tiles based on valueHolder[]→ O(1)
71              else if  valueHolder[5] == 9:→ O(1+1+1+1) → O(1)
72                      valueHolder[4] = valueHolder[5]→ O(1)
73                      valueHolder[5] = temp→ O(1)
74                      moveCounter = moveCounter + 1→ O(1)
75                      update the images of the tiles based on valueHolder[]→ O(1)
76              else if  valueHolder[7] == 9:→ O(1+1+1+1) → O(1)
77                      valueHolder[4] = valueHolder[7]→ O(1)
78                      valueHolder[7] = temp→ O(1)
79                      moveCounter = moveCounter + 1→ O(1)
80                      update the images of the tiles based on valueHolder[]→ O(1)
81      If the player clicks the sixth tile:→ O(1+1) → O(1)
82              set temp = the valueHolder[5]→ O(1)
83              if  valueHolder[4] == 9:→ O(1+1+1+1) → O(1)
84                      valueHolder[5] = valueHolder[4]→ O(1)
85                      valueHolder[4] = temp→ O(1)
86                      moveCounter = moveCounter + 1→ O(1)
87                      update the images of the tiles based on valueHolder[]→ O(1)
88              else if  valueHolder[2] == 9:→ O(1+1+1+1) → O(1)
89                      valueHolder[5] = valueHolder[2]→ O(1)
90                      valueHolder[2] = temp→ O(1)
91                      moveCounter = moveCounter + 1→ O(1)
92                      update the images of the tiles based on valueHolder[]→ O(1)
93              else if  valueHolder[8] == 9:→ O(1+1+1+1) → O(1)
94                      valueHolder[5] = valueHolder[8]→ O(1)
```

95 valueHolder[8] = temp→ **O(1)**

96 moveCounter = moveCounter + 1→ **O(1)**

97 update the images of the tiles based on valueHolder[]→ **O(1)**

98 If the player clicks the seventh tile:→ **O(1+1) → O(1)**

99 set temp = the valueHolder[6]→ **O(1)**

100 if valueHolder[3] == 9:→ **O(1+1+1+1) → O(1)**

101 valueHolder[6] = valueHolder[3]→ **O(1)**

102 valueHolder[3] = temp→ **O(1)**

103 moveCounter = moveCounter + 1→ **O(1)**

104 update the images of the tiles based on valueHolder[]→ **O(1)**

105 else if valueHolder[7] == 9:→ **O(1+1+1+1) → O(1)**

106 valueHolder[6] = valueHolder[7]→ **O(1)**

107 valueHolder[7] = temp→ **O(1)**

108 moveCounter = moveCounter + 1→ **O(1)**

109 update the images of the tiles based on valueHolder[]→ **O(1)**

110 If the player clicks the eighth tile:→ **O(1+1) → O(1)**

111 set temp = the valueHolder[7]→ **O(1)**

112 if valueHolder[6] == 9:→ **O(1+1+1+1) → O(1)**

113 valueHolder[6] = valueHolder[3]→ **O(1)**

114 valueHolder[3] = temp→ **O(1)**

115 moveCounter = moveCounter + 1→ **O(1)**

116 update the images of the tiles based on valueHolder[]→ **O(1)**

117 else if valueHolder[4] == 9:→ **O(1+1+1+1) → O(1)**

118 valueHolder[6] = valueHolder[7]→ **O(1)**

119 valueHolder[7] = temp→ **O(1)**

120 moveCounter = moveCounter + 1→ **O(1)**

121 update the images of the tiles based on valueHolder[]→ **O(1)**

122 else if valueHolder[8] == 9:→ **O(1+1+1+1) → O(1)**

123 valueHolder[7] = valueHolder[8]→ **O(1)**

124 valueHolder[8] = temp→ **O(1)**

125 moveCounter = moveCounter + 1→ **O(1)**

126 update the images of the tiles based on valueHolder[]→ **O(1)**

127 If the player clicks the ninth tile:→ **O(1+1) → O(1)**

128 set temp = the valueHolder[8]→ **O(1)**

129 if valueHolder[7] == 9:→ **O(1+1+1+1) → O(1)**

130 valueHolder[8] = valueHolder[7]→ **O(1)**

131 valueHolder[7] = temp→ **O(1)**

132 moveCounter = moveCounter + 1→ **O(1)**

133 update the images of the tiles based on valueHolder[]→ **O(1)**

134 else if valueHolder[5] == 9:→ **O(1+1+1+1) → O(1)**

135 valueHolder[8] = valueHolder[5]→ **O(1)**

136 valueHolder[5] = temp→ **O(1)**

137 moveCounter = moveCounter + 1→ **O(1)**

138 update the images of the tiles based on valueHolder[]→ **O(1)**

End

If we add the total operations of the algorithm, we can come up with O(1+1+1+1+1+1+1+1+1) = O(9). Since 9 is constant, we can say that the time complexity of the algorithm is O(1).

Here is the algorithm for shuffling the valueHolder array:

Algorithm shuffle ()

Start

1 while i<the length of the valueHolder[]: → **O(3*n)→ O(3n) → O(n)**

2 set integer s = i + (Math.random*valueHolder's length - i)→ **O(1)**

3 set integer temp = valueHolder[s]→ **O(1)**

4 valueHolder[i] = temp→ **O(1)**

End

Since 3 is negligible against n as n grows large, we can say that the time complexity of the algorithm is O(n).