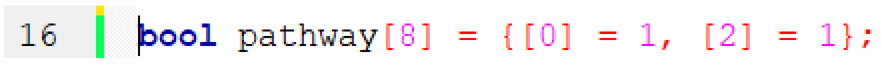
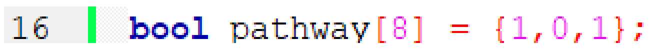
1.

a. Revise line 16 such that you use a designated initializer to set pathways 0 and 2 to true, and the rest will be false. Make the initializer as short as possible.



```
16  bool pathway[8] = {[0] = 1, [2] = 1};
```

b. Revise line 16 such that the initializer will be short as possible (without using a designated initializer)



```
16  bool pathway[8] = {1,0,1};
```

2.

My solution for this problem was to first create a 9x9 array made of characters ABCDEFG and 1s and 0s. Next solution was to ask the user which point they are in and based on the number that they input; this will be the basis for the For loop. This For loop will be used to access a specific row in the array, and then it will loop through each element of the row to see if these elements contain ‘1’ or ‘0’. If element is 0, that means user can’t pass through there so the program will continue until it finds a ‘1’. The program will then assess the column the ‘1’ is located in to check the path to the charging station the user can follow.

My takeaway from this assignment is that it is quite hard and tedious to come up with a decent program that can successfully path through the array. The parts that did not work as intended were at first using a Boolean array and then a separate array for letters A to H. It made the code look quite messy compared to using just characters for the array.  I also think that specific pathing algorithms would be of much better use compared to using for loops because for loops made things confusing and one-dimensional. Finding an element equal to ‘1’ using for loops, one would already need to know which stations are accessible and which are not, compared to an algorithm where it can find a path without pre-existing knowledge of the paths in a given area. Adding on to this, knowing which stations are accessible and which are not, we can easily use switch statements to get the location of the user and from there, just connect each station to each other.