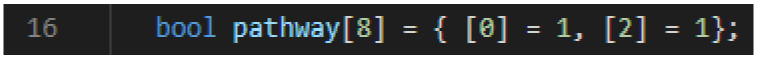
Lecture 6-7

Rainer T. Mayagma
202110417

https://github.com/R4inMeka/CMSC21/tree/main/Lecture6-7

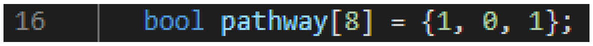
1. A.



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

I used designations here to shorten the code.

B.



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

I assigned values to the first three elements of the array. In C programming, unassigned value of a Boolean is False in default.

2.

My algorithm for number 2:

1. Define the size of the matrix.
2. Initialized the matrix.
3. Print the column titles using for-loop.
4. Print the matrix using for-loop while printing row titles.
5. Get the user input of the starting point.
6. Determining the accessible roads by using if-condition, if matrix[x][starting point] == 1, then road is True/Accessible.
7. Using conditions if point arrived at charging point, loop stops.
8. Print the final destination point.

My approach for finding the accessible roads or True values is from column to row. Just like the example in the assignment guide, the checking for the True values is from the starting point (user input) to i → size of the matrix. This is called Adjacency Matrix. If matrix[x][starting point] == 1, (x is from 0 – 7) then that road is accessible and x will be the new starting point. For G, a→g = 0 but g → a = 1 since there is a one-way path from point g to point a, so I made a special case for this using if conditions if starting point is from G since my approach is from column to row.
However, even if I got to the nearest charging station, the iteration always start at i = 1. If the user input is 5 which is F, the point should not go back to A since C is accessible to F without going back to A.