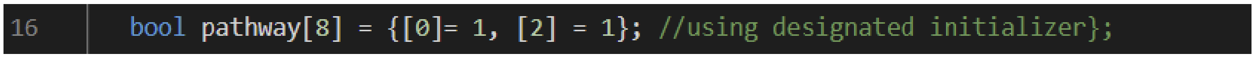
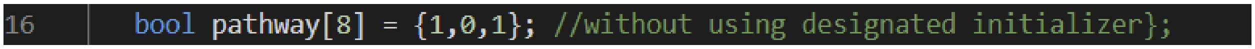
MARJELEN G. MADRIGAL CMSC 21
BS Computer Science 1

1. **A.**



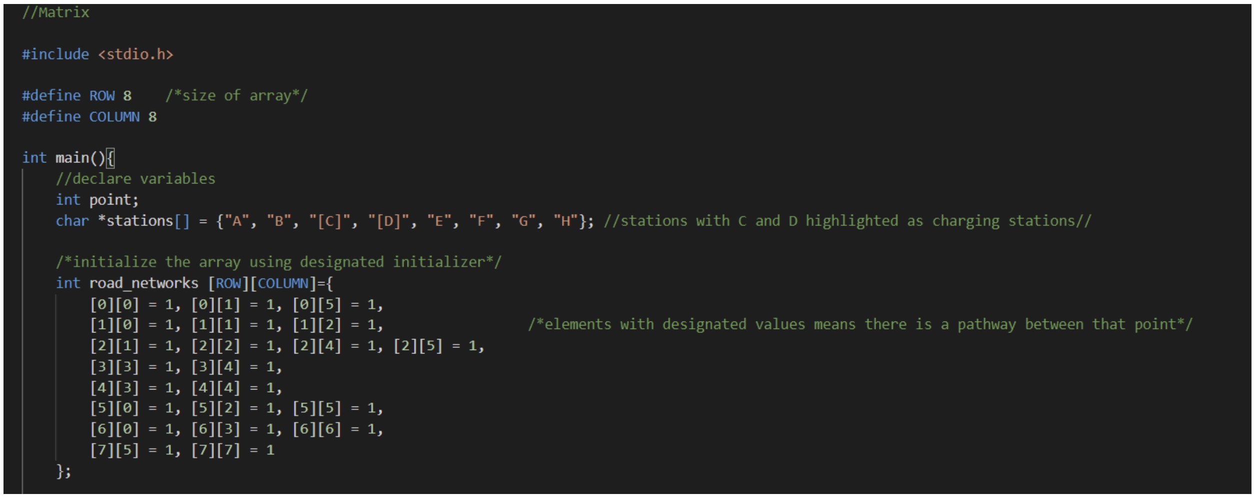
```
bool pathway[8] = {[0]= 1, [2] = 1}; //using designated initializer};
```

B.



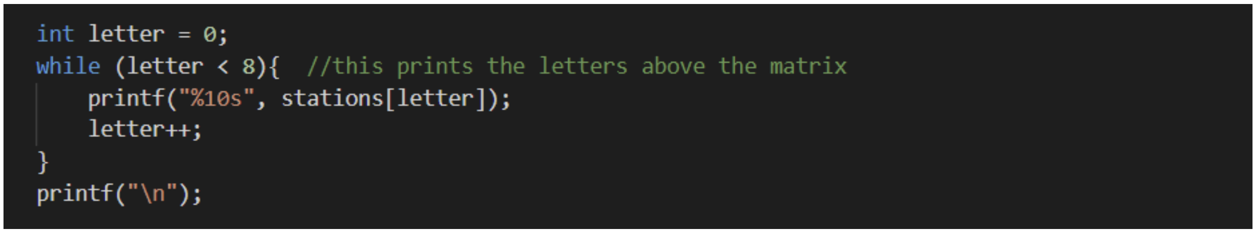
```
bool pathway[8] = {1,0,1}; //without using designated initializer};
```

2. My solution in this problem is not the shortest nor the most optimized route. I applied what I learned from the examples and the PowerPoint presentation. I used a macro to set the size of the matrix into 8 since there are 8 stations, I did not include station I since it's not included in the matrix example. Then I declare the stations as variables with C and D highlighted because they are charging stations. Then using designated initializer method, I initialize the elements of the array which have a value of 1 or meaning they have a pathway.



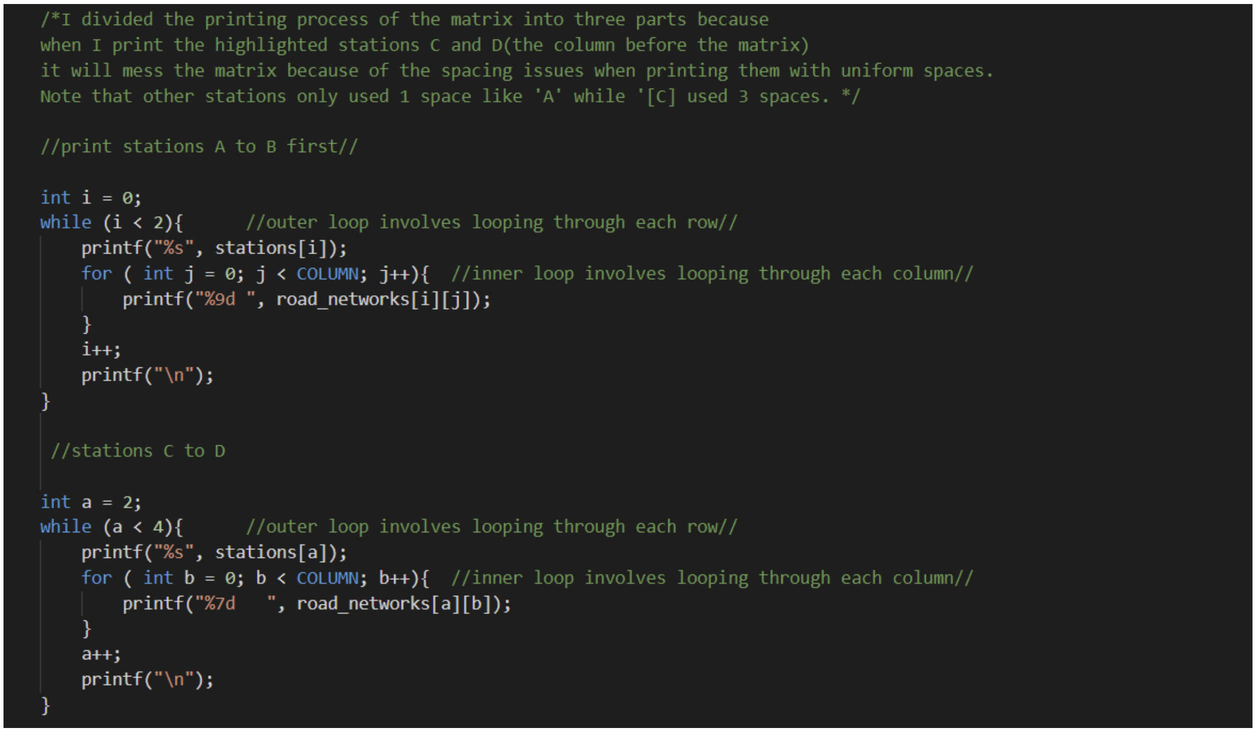
```
//Matrix

#include <stdio.h>

#define ROW 8    /*size of array*/
#define COLUMN 8

int main(){
    //declare variables
    int point;
    char *stations[] = {"A", "B", "[C]", "[D]", "E", "F", "G", "H"}; //stations with C and D highlighted as charging stations//

    /*initialize the array using designated initializer*/
    int road_networks [ROW][COLUMN]={
        [0][0] = 1, [0][1] = 1, [0][5] = 1,
        [1][0] = 1, [1][1] = 1, [1][2] = 1,                /*elements with designated values means there is a pathway between that point*/
        [2][1] = 1, [2][2] = 1, [2][4] = 1, [2][5] = 1,
        [3][3] = 1, [3][4] = 1,
        [4][3] = 1, [4][4] = 1,
        [5][0] = 1, [5][2] = 1, [5][5] = 1,
        [6][0] = 1, [6][3] = 1, [6][6] = 1,
        [7][5] = 1, [7][7] = 1
    };
```

Next, I printed the letters above the matrix.

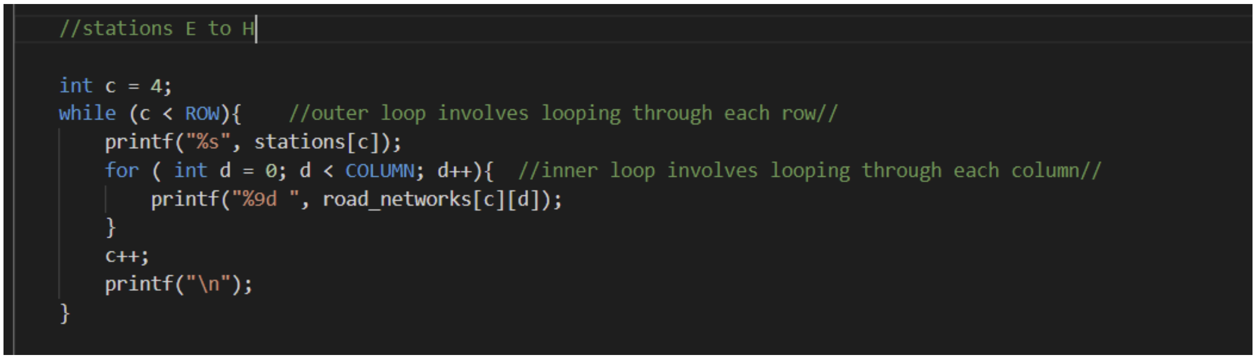


```
int letter = 0;
while (letter < 8){  //this prints the letters above the matrix
    printf("%10s", stations[letter]);
    letter++;
}
printf("\n");
```

Then I divided printing the matrix into three parts. If I used only 1 set of loop the matrix will be messed up and will not align. The while loop loops through each row while the for-loop loops through each column.

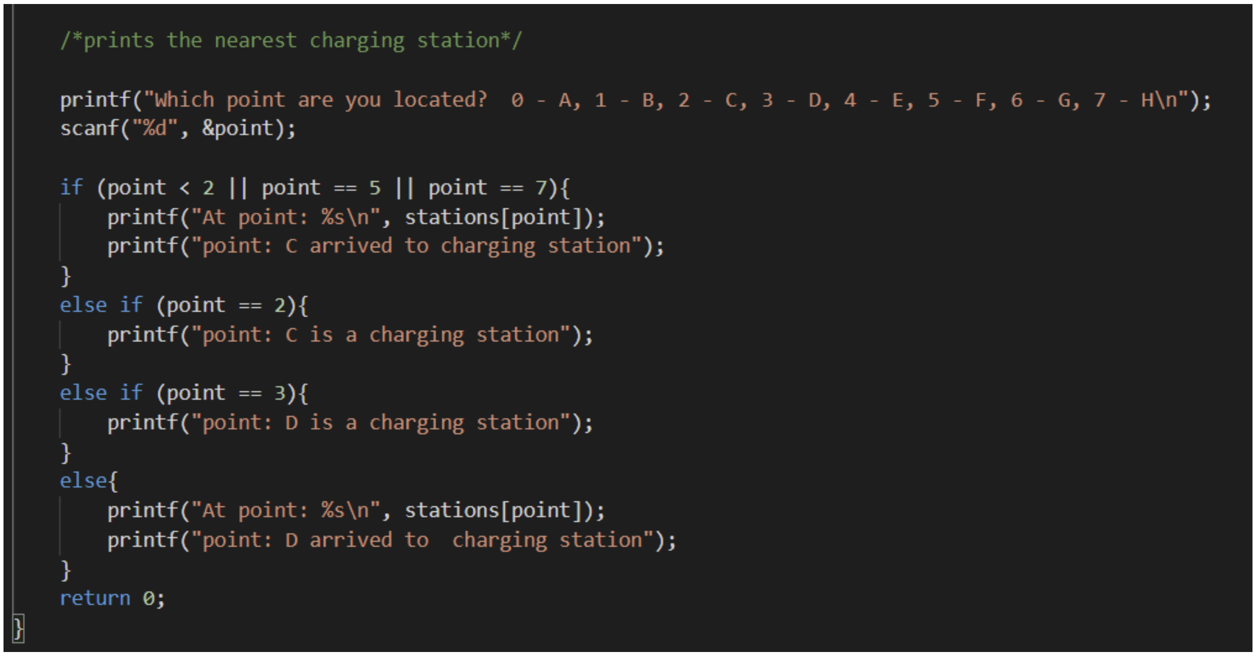


```
/*I divided the printing process of the matrix into three parts because
when I print the highlighted stations C and D(the column before the matrix)
it will mess the matrix because of the spacing issues when printing them with uniform spaces.
Note that other stations only used 1 space like 'A' while '[C] used 3 spaces. */

//print stations A to B first//

int i = 0;
while (i < 2){      //outer loop involves looping through each row//
    printf("%s", stations[i]);
    for ( int j = 0; j < COLUMN; j++){  //inner loop involves looping through each column//
        printf("%9d ", road_networks[i][j]);
    }
    i++;
    printf("\n");
}

 //stations C to D

int a = 2;
while (a < 4){      //outer loop involves looping through each row//
    printf("%s", stations[a]);
    for ( int b = 0; b < COLUMN; b++){  //inner loop involves looping through each column//
        printf("%7d   ", road_networks[a][b]);
    }
    a++;
    printf("\n");
}
```



```
//stations E to H

int c = 4;
while (c < ROW){     //outer loop involves looping through each row//
    printf("%s", stations[c]);
    for ( int d = 0; d < COLUMN; d++){  //inner loop involves looping through each column//
        printf("%9d ", road_networks[c][d]);
    }
    c++;
    printf("\n");
}
```

Then I used the illustration from the instruction to determine what is the nearest charging station if a point is given. Points A to B, F, & H have C as their charging station, while point G have D as its charging station. Points C and D is a charging station themselves.



```
    /*prints the nearest charging station*/

    printf("Which point are you located?  0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H\n");
    scanf("%d", &point);

    if (point < 2 || point == 5 || point == 7){
        printf("At point: %s\n", stations[point]);
        printf("point: C arrived to charging station");
    }
    else if (point == 2){
        printf("point: C is a charging station");
    }
    else if (point == 3){
        printf("point: D is a charging station");
    }
    else{
        printf("At point: %s\n", stations[point]);
        printf("point: D arrived to  charging station");
    }
    return 0;
}
```

Github Link: https://github.com/marj12/CMSC21/tree/main/Lecture6-7/Assignments