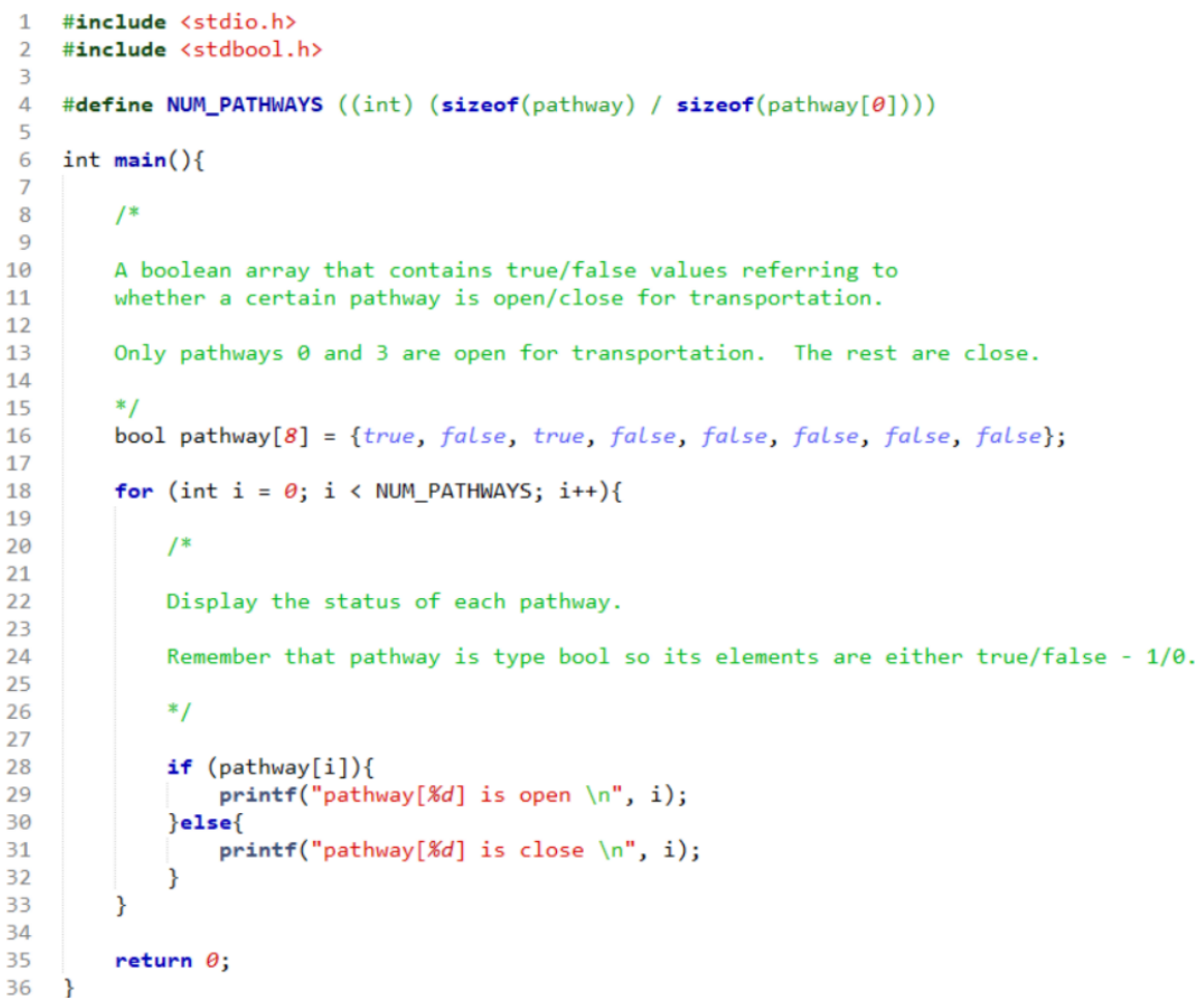
Christian Justin J. Salinas – 2021-50521
Instructor Ara Abigail Ambita
CMSC 21 – 2

Lectures 6 - 7 Assignment

1. In the program below, an array named pathway contains eight bool values. Each bool element refers to whether a pathway is open or close for transportation.

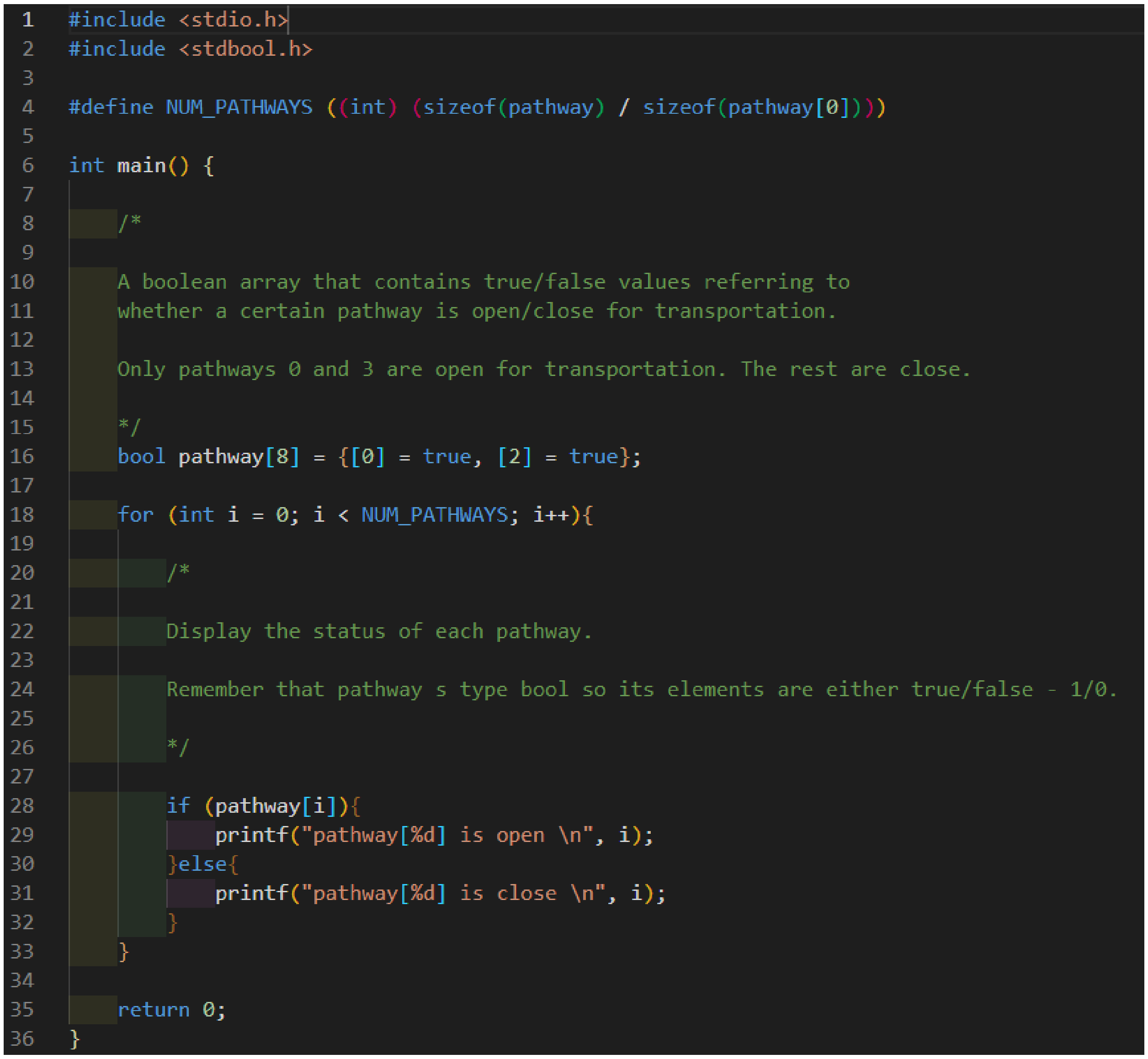
Only pathways 0 and 2 are open while the rest are still close due to road constructions and fixings.



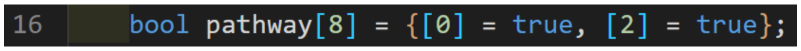
```c
#include <stdio.h>
#include <stdbool.h>

#define NUM_PATHWAYS ((int) (sizeof(pathway) / sizeof(pathway[0])))

int main(){

    /*

    A boolean array that contains true/false values referring to
    whether a certain pathway is open/close for transportation.

    Only pathways 0 and 3 are open for transportation.  The rest are close.

    */
    bool pathway[8] = {true, false, true, false, false, false, false, false};

    for (int i = 0; i < NUM_PATHWAYS; i++){

        /*

        Display the status of each pathway.

        Remember that pathway is type bool so its elements are either true/false - 1/0.

        */

        if (pathway[i]){
            printf("pathway[%d] is open \n", i);
        }else{
            printf("pathway[%d] is close \n", i);
        }
    }

    return 0;
}
```

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.

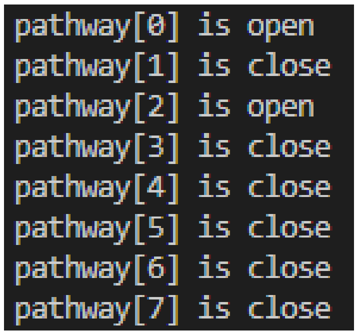
Complete source code:



```c
1   #include <stdio.h>
2   #include <stdbool.h>
3
4   #define NUM_PATHWAYS ((int) (sizeof(pathway) / sizeof(pathway[0])))
5
6   int main() {
7
8       /*
9
10      A boolean array that contains true/false values referring to
11      whether a certain pathway is open/close for transportation.
12
13      Only pathways 0 and 3 are open for transportation. The rest are close.
14
15      */
16      bool pathway[8] = {[0] = true, [2] = true};
17
18      for (int i = 0; i < NUM_PATHWAYS; i++){
19
20          /*
21
22          Display the status of each pathway.
23
24          Remember that pathway s type bool so its elements are either true/false - 1/0.
25
26          */
27
28          if (pathway[i]){
29              printf("pathway[%d] is open \n", i);
30          }else{
31              printf("pathway[%d] is close \n", i);
32          }
33      }
34
35      return 0;
36  }
```

On line 16: 

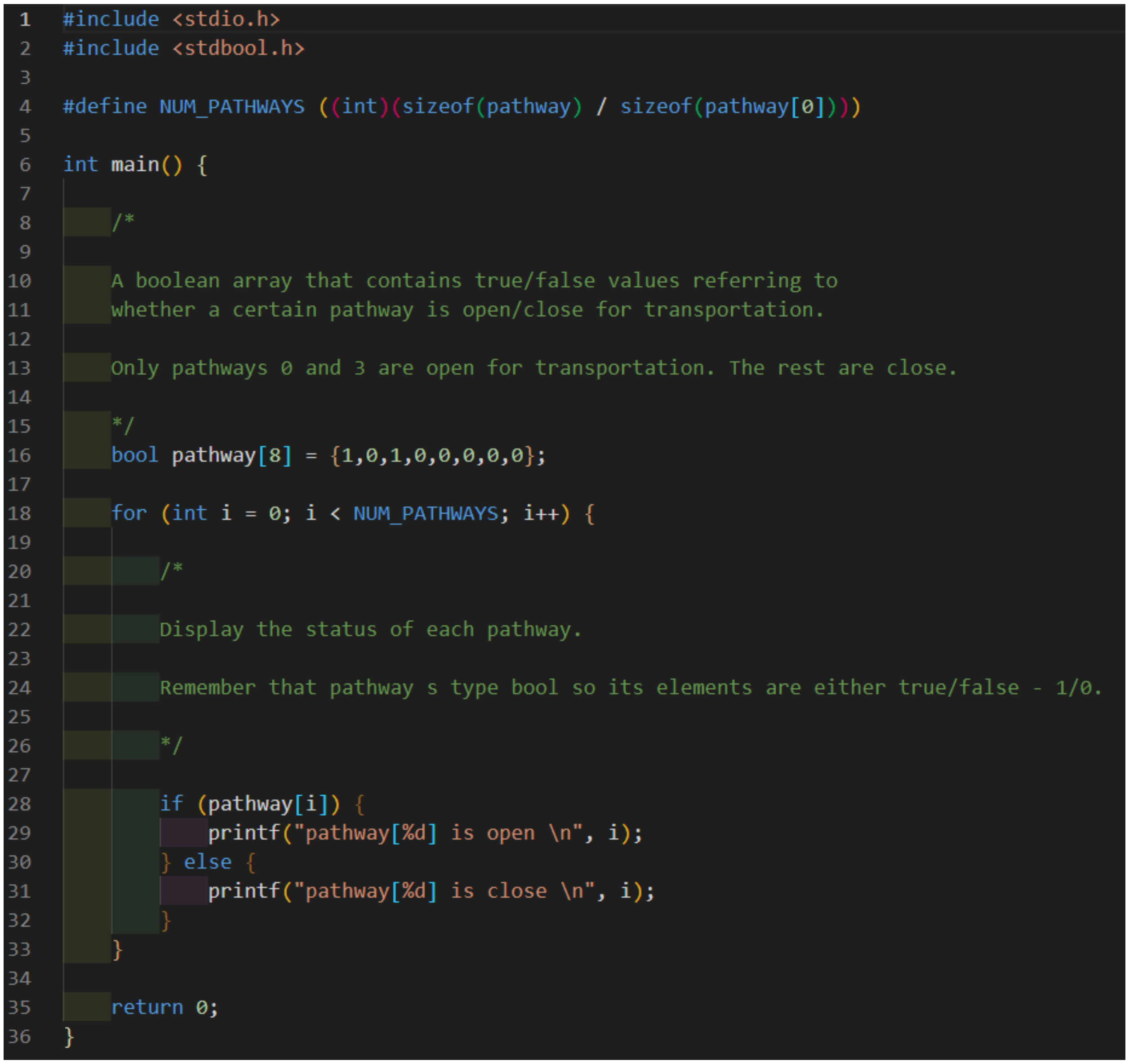
```c
16      bool pathway[8] = {[0] = true, [2] = true};
```

Output: 

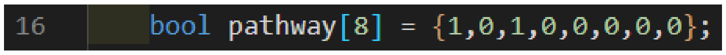
```
pathway[0] is open
pathway[1] is close
pathway[2] is open
pathway[3] is close
pathway[4] is close
pathway[5] is close
pathway[6] is close
pathway[7] is close
```

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

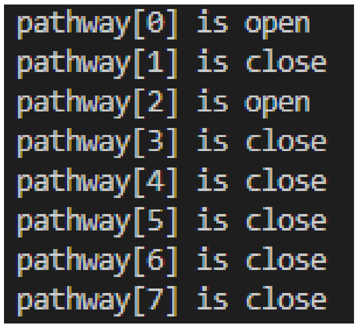
Complete source code:



```
1   #include <stdio.h>
2   #include <stdbool.h>
3
4   #define NUM_PATHWAYS ((int)(sizeof(pathway) / sizeof(pathway[0])))
5
6   int main() {
7
8       /*
9
10      A boolean array that contains true/false values referring to
11      whether a certain pathway is open/close for transportation.
12
13      Only pathways 0 and 3 are open for transportation. The rest are close.
14
15      */
16      bool pathway[8] = {1,0,1,0,0,0,0,0};
17
18      for (int i = 0; i < NUM_PATHWAYS; i++) {
19
20          /*
21
22          Display the status of each pathway.
23
24          Remember that pathway s type bool so its elements are either true/false - 1/0.
25
26          */
27
28          if (pathway[i]) {
29              printf("pathway[%d] is open \n", i);
30          } else {
31              printf("pathway[%d] is close \n", i);
32          }
33      }
34
35      return 0;
36  }
```

On line 16: 

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

Output: 

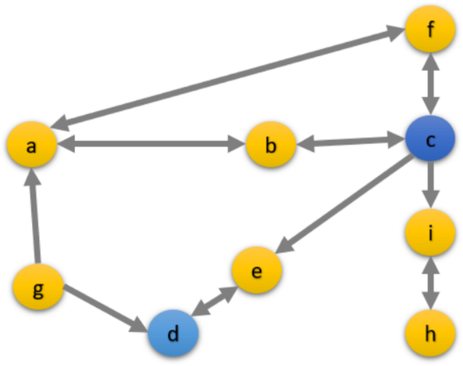
```
pathway[0] is open
pathway[1] is close
pathway[2] is open
pathway[3] is close
pathway[4] is close
pathway[5] is close
pathway[6] is close
pathway[7] is close
```

2. A road network can be represented using graphs. Assuming we have points / stations a, b, c, d, e, f, g, and h, we can represent a direct path from a point to another point using arrows.

For example, based on the graph below:

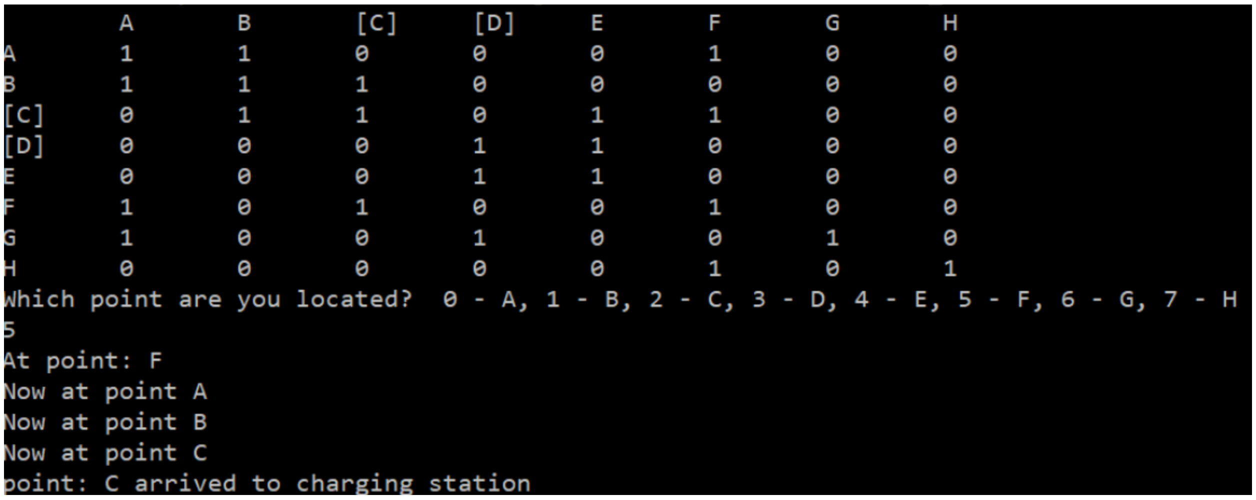
- There is a two-way path between point a and point b, point a and point f, point f and point c, and point d and e
- There is a one-way path from point c to point i but no direct path between point i to point c.

All of the nodes are points/destinations, but the yellow ones specifically represent charging stations. The road network between these points/destinations can be represented using an adjacency matrix of Booleans (0s and 1s), as shown below. For instance, a → b = 1 and b → a = 1 given that there's a two-way direct path between a and b. Meanwhile, a → c = 0 since there is no direct path between a and c. Moreover, a → g = 0 but g → a = 1 since there is a one-way path from point g to point a.



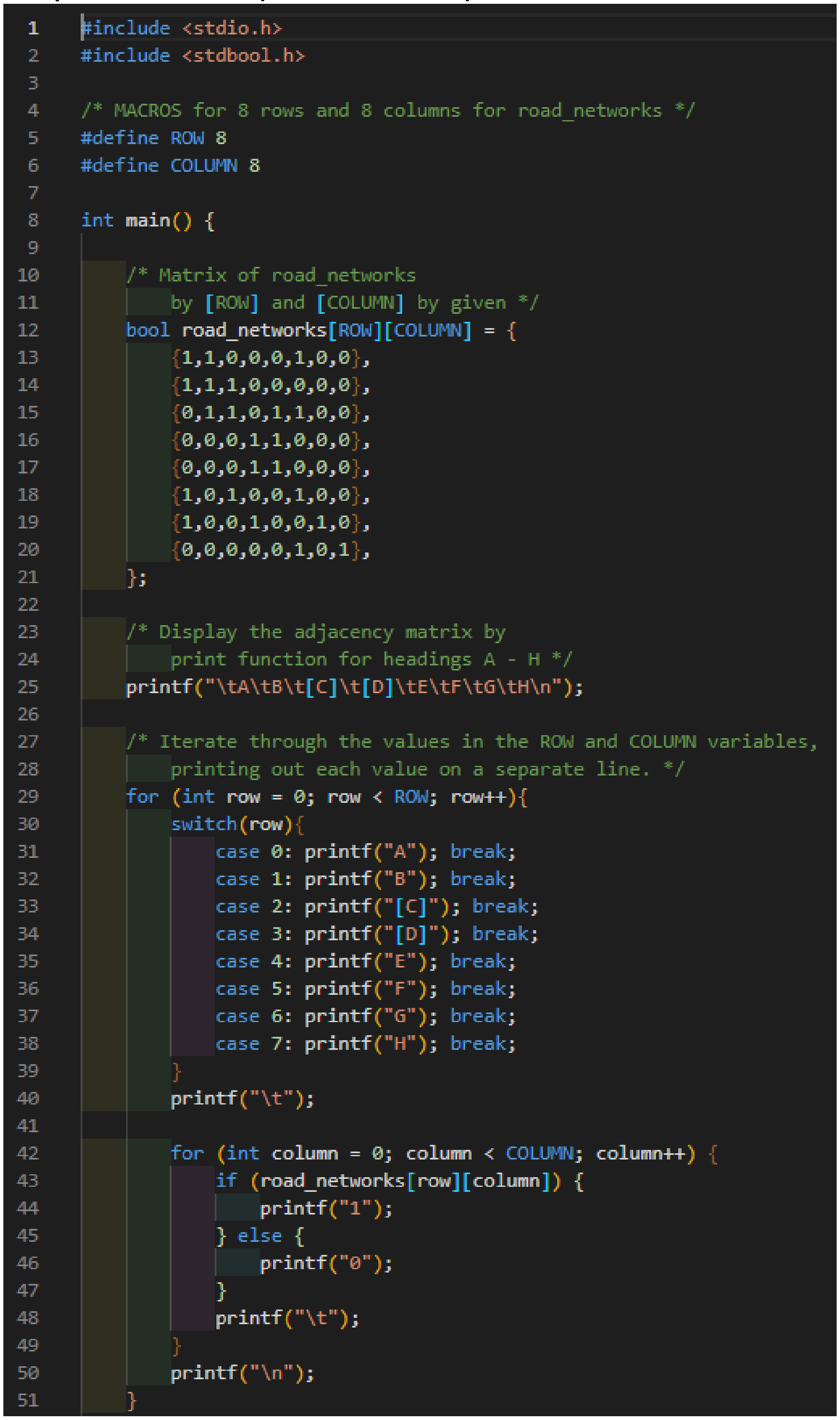
	a	b	c	d	e	f	g	h
a	1	1	0	0	0	1	0	0
b	1	1	1	0	0	0	0	0
c	0	1	1	0	1	1	0	0
d	0	0	0	1	1	0	0	0
e	0	0	0	1	1	0	0	0
f	1	0	1	0	0	1	0	0
g	1	0	0	1	0	0	1	0
h	0	0	0	0	0	1	0	1

As a programming assignment:
1. Declare and initialize a road_networks multidimensional array that represents the adjacency matrix
2. Display the adjacency matrix. Put a bracket to the points/destinations that are considered as charging stations, e.g. [c], [d]
3. Given a point / destination, determine the nearest charging station. For example, if you are in point a, the nearest charging station is point c. If you are in point e, the nearest charging station is point d.
4. Bonus: Use a macro to define the size of the 2d array

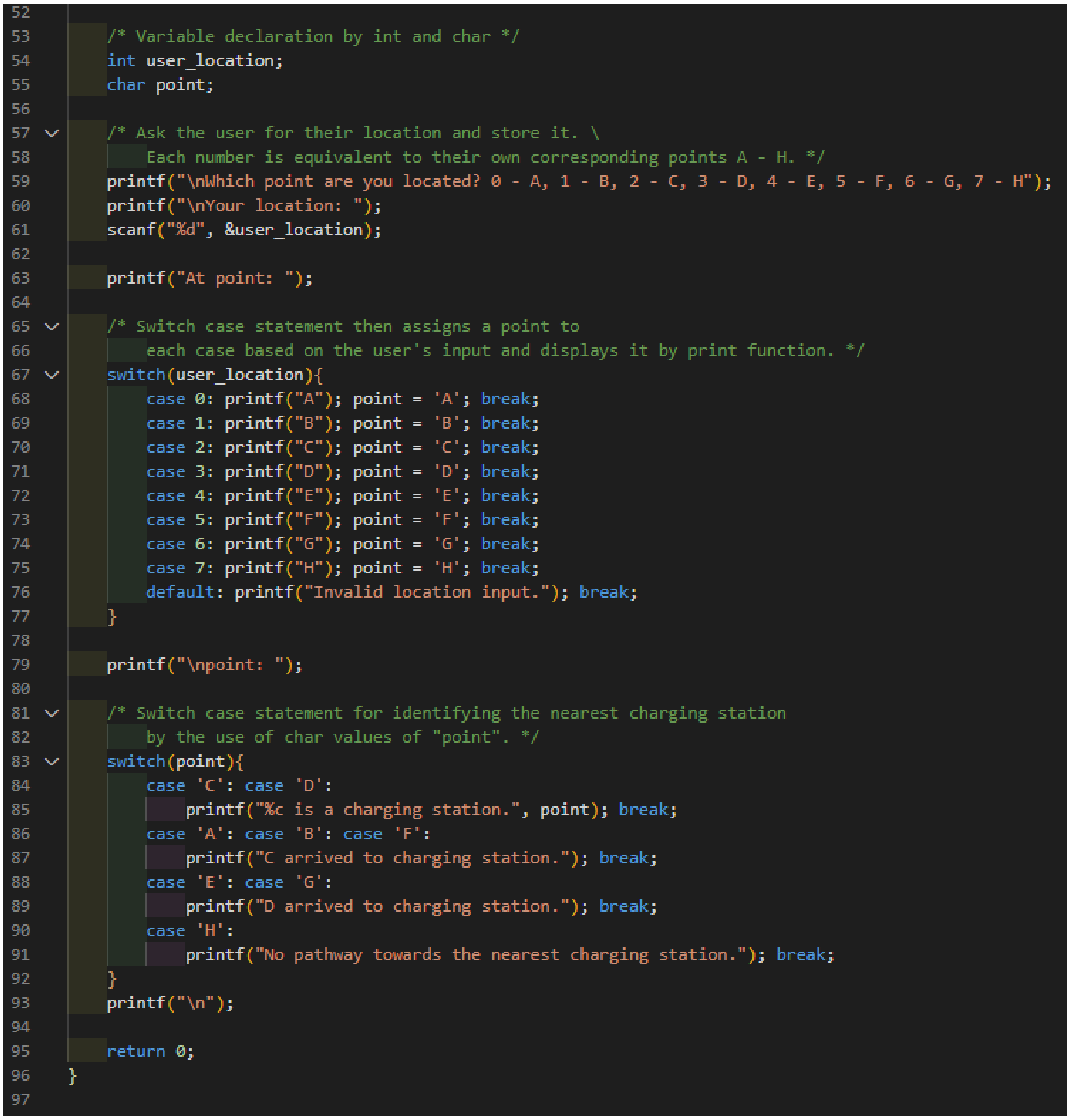


```
        A       B       [C]     [D]     E       F       G       H
A       1       1       0       0       0       1       0       0
B       1       1       1       0       0       0       0       0
[C]     0       1       1       0       1       1       0       0
[D]     0       0       0       1       1       0       0       0
E       0       0       0       1       1       0       0       0
F       1       0       1       0       0       1       0       0
G       1       0       0       1       0       0       1       0
H       0       0       0       0       0       1       0       1
Which point are you located?  0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H
5
At point: F
Now at point A
Now at point B
Now at point C
point: C arrived to charging station
```

Complete source code (without functions):



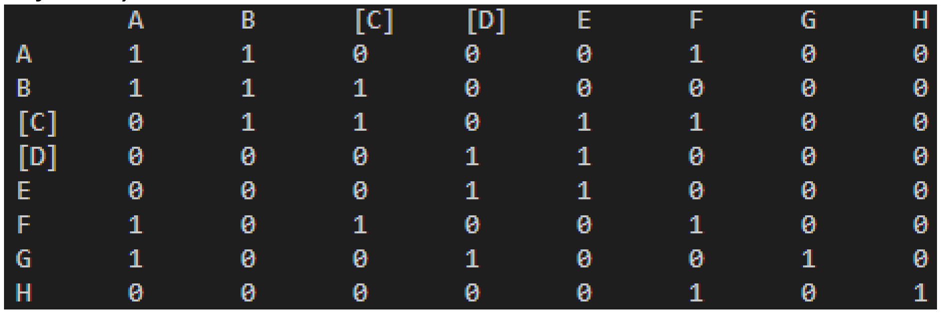
```c
#include <stdio.h>
#include <stdbool.h>

/* MACROS for 8 rows and 8 columns for road_networks */
#define ROW 8
#define COLUMN 8

int main() {

    /* Matrix of road_networks
        by [ROW] and [COLUMN] by given */
    bool road_networks[ROW][COLUMN] = {
        {1,1,0,0,0,1,0,0},
        {1,1,1,0,0,0,0,0},
        {0,1,1,0,1,1,0,0},
        {0,0,0,1,1,0,0,0},
        {0,0,0,1,1,0,0,0},
        {1,0,1,0,0,1,0,0},
        {1,0,0,1,0,0,1,0},
        {0,0,0,0,0,1,0,1},
    };

    /* Display the adjacency matrix by
        print function for headings A - H */
    printf("\tA\tB\t[C]\t[D]\tE\tF\tG\tH\n");

    /* Iterate through the values in the ROW and COLUMN variables,
        printing out each value on a separate line. */
    for (int row = 0; row < ROW; row++){
        switch(row){
            case 0: printf("A"); break;
            case 1: printf("B"); break;
            case 2: printf("[C]"); break;
            case 3: printf("[D]"); break;
            case 4: printf("E"); break;
            case 5: printf("F"); break;
            case 6: printf("G"); break;
            case 7: printf("H"); break;
        }
        printf("\t");

        for (int column = 0; column < COLUMN; column++) {
            if (road_networks[row][column]) {
                printf("1");
            } else {
                printf("0");
            }
            printf("\t");
        }
        printf("\n");
    }
```



```c

    /* Variable declaration by int and char */
    int user_location;
    char point;

    /* Ask the user for their location and store it. \
        Each number is equivalent to their own corresponding points A - H. */
    printf("\nWhich point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H");
    printf("\nYour location: ");
    scanf("%d", &user_location);

    printf("At point: ");

    /* Switch case statement then assigns a point to
        each case based on the user's input and displays it by print function. */
    switch(user_location){
        case 0: printf("A"); point = 'A'; break;
        case 1: printf("B"); point = 'B'; break;
        case 2: printf("C"); point = 'C'; break;
        case 3: printf("D"); point = 'D'; break;
        case 4: printf("E"); point = 'E'; break;
        case 5: printf("F"); point = 'F'; break;
        case 6: printf("G"); point = 'G'; break;
        case 7: printf("H"); point = 'H'; break;
        default: printf("Invalid location input."); break;
    }

    printf("\npoint: ");

    /* Switch case statement for identifying the nearest charging station
        by the use of char values of "point". */
    switch(point){
        case 'C': case 'D':
            printf("%c is a charging station.", point); break;
        case 'A': case 'B': case 'F':
            printf("C arrived to charging station."); break;
        case 'E': case 'G':
            printf("D arrived to charging station."); break;
        case 'H':
            printf("No pathway towards the nearest charging station."); break;
    }
    printf("\n");

    return 0;
}

```

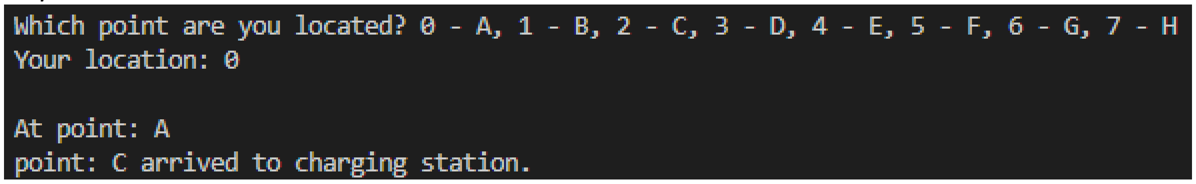
Outputs:

Adjacency Matrix:



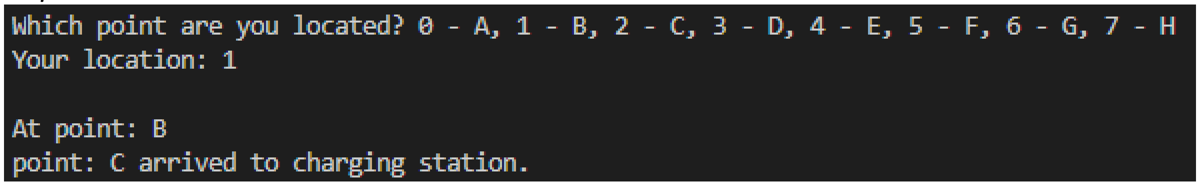
```
        A       B       [C]     [D]     E       F       G       H
A       1       1       0       0       0       1       0       0
B       1       1       1       0       0       0       0       0
[C]     0       1       1       0       1       1       0       0
[D]     0       0       0       1       1       0       0       0
E       0       0       0       1       1       0       0       0
F       1       0       1       0       0       1       0       0
G       1       0       0       1       0       0       1       0
H       0       0       0       0       0       1       0       1
```

At point A:



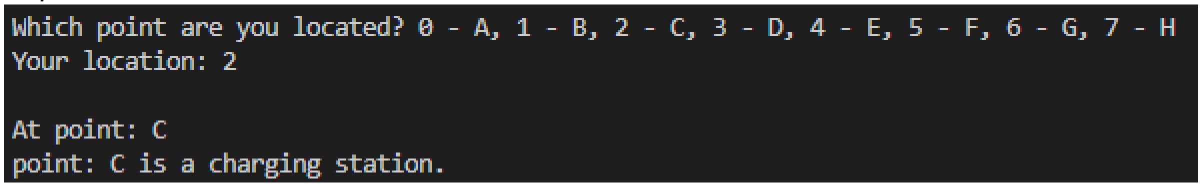
```
Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H
Your location: 0

At point: A
point: C arrived to charging station.
```

At point B:



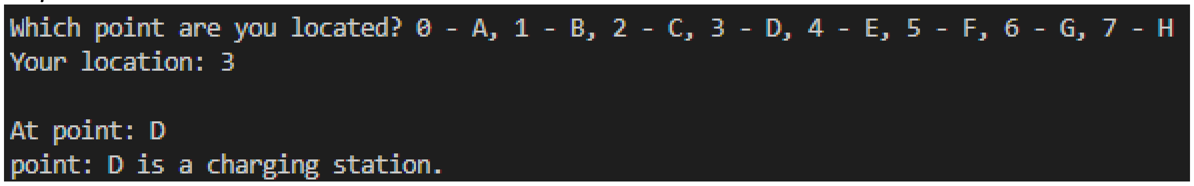
```
Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H
Your location: 1

At point: B
point: C arrived to charging station.
```

At point C:



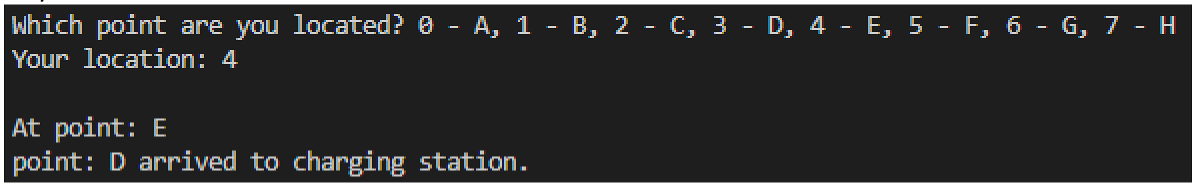
```
Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H
Your location: 2

At point: C
point: C is a charging station.
```

At point D:



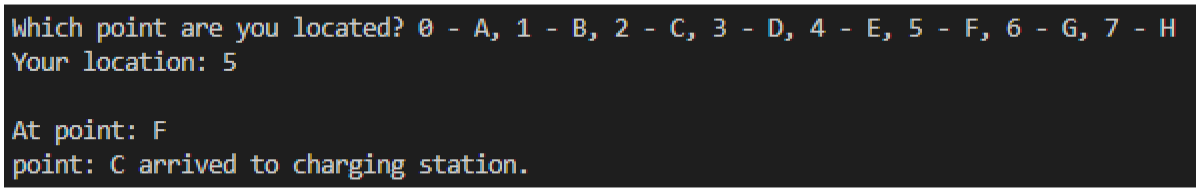
```
Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H
Your location: 3

At point: D
point: D is a charging station.
```

At point E:



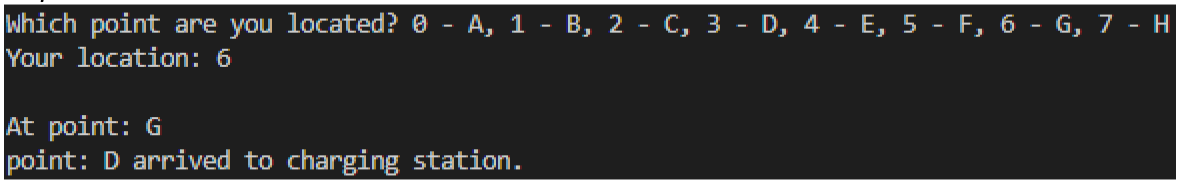
```
Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H
Your location: 4

At point: E
point: D arrived to charging station.
```

At point F:



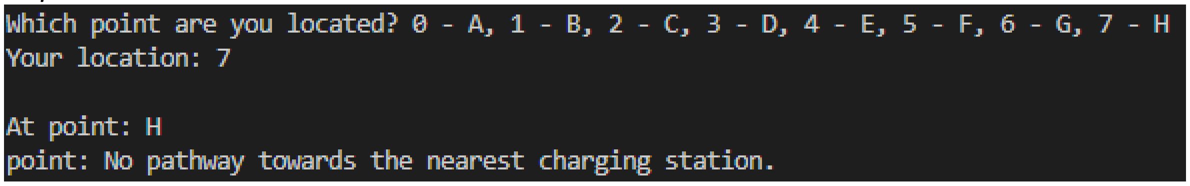
```
Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H
Your location: 5

At point: F
point: C arrived to charging station.
```

At point G:



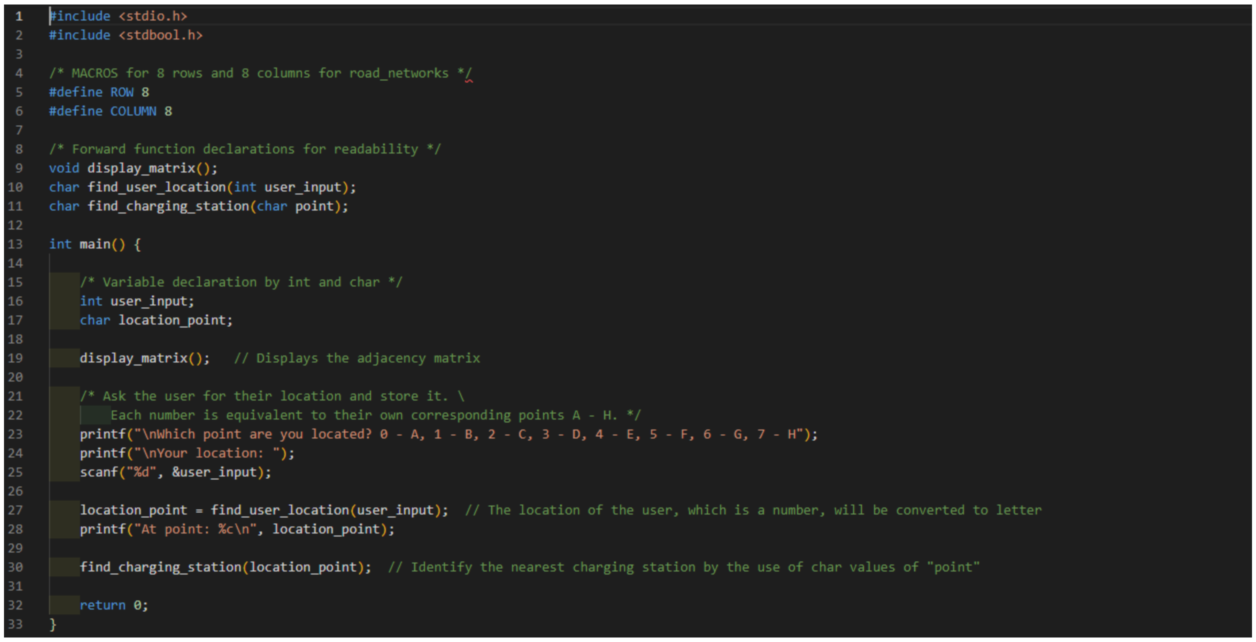
```
Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H
Your location: 6

At point: G
point: D arrived to charging station.
```

At point H:

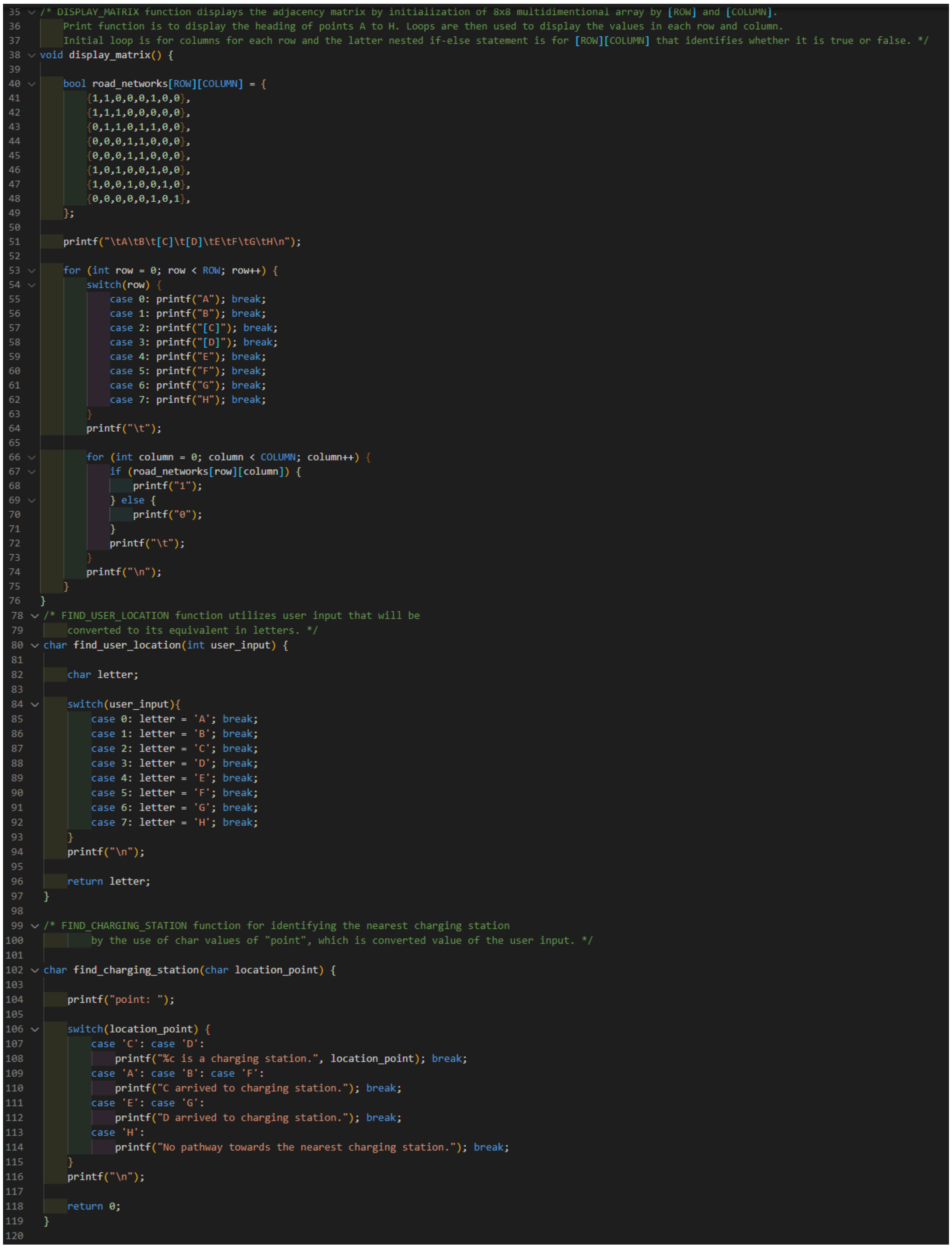


```
Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H
Your location: 7

At point: H
point: No pathway towards the nearest charging station.
```

Complete source code (with functions):



```c
#include <stdio.h>
#include <stdbool.h>

/* MACROS for 8 rows and 8 columns for road_networks */
#define ROW 8
#define COLUMN 8

/* Forward function declarations for readability */
void display_matrix();
char find_user_location(int user_input);
char find_charging_station(char point);

int main() {

    /* Variable declaration by int and char */
    int user_input;
    char location_point;

    display_matrix();   // Displays the adjacency matrix

    /* Ask the user for their location and store it. \
        Each number is equivalent to their own corresponding points A - H. */
    printf("\nWhich point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H");
    printf("\nYour location: ");
    scanf("%d", &user_input);

    location_point = find_user_location(user_input);  // The location of the user, which is a number, will be converted to letter
    printf("At point: %c\n", location_point);

    find_charging_station(location_point);  // Identify the nearest charging station by the use of char values of "point"

    return 0;
}
```



```c
/* DISPLAY_MATRIX function displays the adjacency matrix by initialization of 8x8 multidimentional array by [ROW] and [COLUMN].
    Print function is to display the heading of points A to H. Loops are then used to display the values in each row and column.
    Initial loop is for columns for each row and the latter nested if-else statement is for [ROW][COLUMN] that identifies whether it is true or false. */
void display_matrix() {

    bool road_networks[ROW][COLUMN] = {
        {1,1,0,0,0,1,0,0},
        {1,1,1,0,0,0,0,0},
        {0,1,1,0,1,1,0,0},
        {0,0,0,1,1,0,0,0},
        {0,0,0,1,1,0,0,0},
        {1,0,1,0,0,1,0,0},
        {1,0,0,1,0,0,1,0},
        {0,0,0,0,0,1,0,1},
    };

    printf("\tA\tB\t[C]\t[D]\tE\tF\tG\tH\n");

    for (int row = 0; row < ROW; row++) {
        switch(row) {
            case 0: printf("A"); break;
            case 1: printf("B"); break;
            case 2: printf("[C]"); break;
            case 3: printf("[D]"); break;
            case 4: printf("E"); break;
            case 5: printf("F"); break;
            case 6: printf("G"); break;
            case 7: printf("H"); break;
        }
        printf("\t");

        for (int column = 0; column < COLUMN; column++) {
            if (road_networks[row][column]) {
                printf("1");
            } else {
                printf("0");
            }
            printf("\t");
        }
        printf("\n");
    }
}
/* FIND_USER_LOCATION function utilizes user input that will be
    converted to its equivalent in letters. */
char find_user_location(int user_input) {

    char letter;

    switch(user_input){
        case 0: letter = 'A'; break;
        case 1: letter = 'B'; break;
        case 2: letter = 'C'; break;
        case 3: letter = 'D'; break;
        case 4: letter = 'E'; break;
        case 5: letter = 'F'; break;
        case 6: letter = 'G'; break;
        case 7: letter = 'H'; break;
    }
    printf("\n");

    return letter;
}

/* FIND_CHARGING_STATION function for identifying the nearest charging station
        by the use of char values of "point", which is converted value of the user input. */

char find_charging_station(char location_point) {

    printf("point: ");

    switch(location_point) {
        case 'C': case 'D':
            printf("%c is a charging station.", location_point); break;
        case 'A': case 'B': case 'F':
            printf("C arrived to charging station."); break;
        case 'E': case 'G':
            printf("D arrived to charging station."); break;
        case 'H':
            printf("No pathway towards the nearest charging station."); break;
    }
    printf("\n");

    return 0;
}
```

SOLUTION

The multidimensional array was initialized using the bracketed rows being in bool type. Macros were used to define the size of the 2d array with points being from A to H. Thus, both rows and columns were set to 8. The multidimensional array's values are either true (1) or false (0). "road_network" being the array, it is printed by the print function in a 2d setting.

The first row of points A to H are displayed together with the corresponding rows and columns by utilizing loops. The initial loop identifies and determines the row in which the loop is iterating. Moreover, using a nested loop for the next columns for values (1s or 0s) will be displayed. It determines whether which column is iterating. Finally, a nested if-else statement is used for the value to be displayed. To explain, if a value in a particular [ROW][COLUMN] is true, then it will display "1". Otherwise, "0". Simply, "1" in the adjacency matrix means that that [ROW][COLUMN] value exists.

Furthermore, the program asks the user for an input for their location. The user can accept values of integer 0 to 7 as each number corresponds to a point which is determined using a switch case statement. This will be stored and will be used for the next corresponding switch case statement.

At last, it will accordingly determine the nearest charging station (either C or D). Another switch-case statement checks this.

TAKEAWAYS/INSIGHTS

As each node in the array contains a Boolean value that indicates whether or not that particular road network is connected to another one—this sounds like a real-world situation! The problem given is I believe one of those problems engineers and researchers have to go through in order to give the best possible solution and it surely did make me a lot more curious.

As much as possible, I did most of the best practices I learned from the modules by Ma'am Ara and been having fun so far. Using loops and nested loops to display the adjacency matrix, combination of switch-case statements and if-else statements with conditions accordingly and trying to make the most of the information given in the assignment.

I feel like I can do much more, like adding a step-by-step output process of how the user approaches and determines the way to the nearest point/node (to update the user) but still being as concise. With this assignment focusing on arrays, I had fun learning about it as this showcases an example of a real-world situation. 😊

P.S. I tried using Visual Studio Code for this assignment! Been trying out new IDEs for the sake of learning.

GitHub Link: https://github.com/xkaze09/CMSC21/tree/main/Lecture6-7/Assignments