

CMSC 21 – 1

Assignment #2 – Lecture 6 & 7

1.

a.

```
[*] prac.c [*] #1.a..c #1.b..c
1  #include <stdio.h>
2  #include <stdbool.h>
3
4  #define NUM_PATHWAYS ((int) (sizeof(pathway) / sizeof(pathway[0]))) // define the total number of the array
5
6  int main(){
7      // only pathway[0] & pathway[2] are true. If you did not set a value for the other positions, it is automated to give a zero.
8      bool pathway[8] = {[0] = 1, [2] = 1};
9
10     // show whether the positions in the array are open or closed.
11     for (int i = 0; i < NUM_PATHWAYS; i++){
12         if (pathway[i]){
13             printf("pathway [%d] is open \n", i);
14         }else{
15             printf("pathway [%d] is closed \n", i);
16         }
17     }
18 }
19
```

b.

```
[*] prac.c [*] #1.a..c #1.b..c
1  #include <stdio.h>
2  #include <stdbool.h>
3
4  #define NUM_PATHWAYS ((int) (sizeof(pathway) / sizeof(pathway[0]))) // define the total number of the array
5
6  int main(){
7      // only pathway[0] & pathway[2] are true. If you did not set a value for the other positions, it is automated to give a zero.
8      bool pathway[8] = {1, 0, 1};
9
10     // show whether the positions in the array are open or closed.
11     for (int i = 0; i < NUM_PATHWAYS; i++){
12         if (pathway[i]){
13             printf("pathway [%d] is open \n", i);
14         }else{
15             printf("pathway [%d] is closed \n", i);
16         }
17     }
18 }
19
```

2.

```
1 #include <stdio.h>
2 #define len ((int) (sizeof(letters) / sizeof(letters[0])))
3
4 int main(void){
5
6     // Declare and create an array indicating the road network (or paths where each point can go to)
7     // This will be the "inner" part of the matrix
8     int road_networks [9][9] = {{[1][1] = 1, [1][2] = 1, [1][6] = 1,
9                                  [2][1] = 1, [2][2] = 1, [2][3] = 1,
10                                 [3][2] = 1, [3][3] = 1, [3][5] = 1, [3][6] = 1,
11                                 [4][4] = 1, [4][5] = 1,
12                                 [5][4] = 1, [5][5] = 1,
13                                 [6][1] = 1, [6][3] = 1, [6][6] = 1,
14                                 [7][1] = 1, [7][4] = 1, [7][7] = 1,
15                                 [8][6] = 1, [8][8] = 1 }};
16
17     // Put the Letters (the "points") in an array as well.
18     // This will be the "outer" part of the matrix
19     char letters [9][9] = {[0][1] = 'A', [0][2] = 'B', [0][3] = 'C', [0][4] = 'D', [0][5] = 'E', [0][6] = 'F', [0][7] = 'G', [0][8] = 'H',
20                             [1][0] = 'A', [2][0] = 'B', [3][0] = 'C', [4][0] = 'D', [5][0] = 'E', [6][0] = 'F', [7][0] = 'G', [8][0] = 'H'};
21
22     int i, j; // declare the variables that will be used for the rows and columns of the array, respectively.
23
24     // printing of the adjacency matrix
25     for (i = 0; i < len; i++){
26         for (j = 0; j < len; j++){
27
28             if((j == 3) && (i == 0) || (j == 4) && (i == 0)){ // To insert the brackets for stations C & D in the 0th row
29                 printf("[%c]\t", letters[i][j]); // Indicates that the point is a charging station
30             }
31             else if ((i == 3) && (j == 0) || (i == 4) && (j == 0)){ // To insert the brackets for stations C & D in the 0th column
32                 printf("[%c]\t", letters[i][j]); // Indicates that the point is a charging station
33             }
34             else if ((i == 0) || (j == 0)){ // Printing the rest of the points
35                 printf("%c\t", letters[i][j]);
36             }
37             else{
38                 printf("%d\t", road_networks[i][j]); // Printing of the road networks
39             }
40
41             if (j == 8){ // The next print will be on the next line
42                 printf("\n"); // For design purposes
43             }
44         }
45     }
46
47     // Ask user his/her current Location
48     printf("Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H\n");
49     scanf("%d", &j);
50
51     char location = letters[0][j+1];
52
53     if ((j == 2) || (j == 3)){ // For the case of point C & D, which is already a charging station.
54         printf("point: %c is a charging station", location);
55     }
56     else{
57         // For the cases of A, B, E, F, G, & H
58         printf("At point: %c \n", location); // Print user's current location
59
60         switch(location){
61             // For the case of A, which is the only point that needs to get to another point before reaching a charging station
62             // The points where C is the charging station
63             case 'A':
64                 printf("Now at point: B\n");
65             case 'B':
66             case 'F':
67                 printf("point: C is a charging station");
68                 break;
69
70             // The points where D is the charging station
71             case 'E':
72             case 'G':
73                 printf("point: D is a charging station");
74                 break;
75
76             // As for point H, it cannot go to either C or D.
77             case 'H':
78                 printf("No path to a charging station");
79             }
80     }
81 }
```

Because of item #1, I was reminded how to deal with arrays that have boolean as their values. You will only need to specify the positions of the 1s since the rest will be considered as 0s. No need to further type everything, which is really convenient.

The next array is for the “points”. I made an array for this so that it will be easier when coding for the printing of the matrix. If it were not for the array, I might use ifs statements for each letter (which can be a hassle).

A thing that did not work as intended was the paths (where each point can or cannot go to). Even though there is a matrix shown, I was not able to utilize it (the array with boolean values) well when coding for the part where the program tells the user the nearest charging station. Instead, I took note of where each point's nearing station is.

Github link:

<https://github.com/selenabarrameda/CMSC-21/tree/main/Lecture%206-7/Assignments>