CMSC 21

Lecture 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.

   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 (`Source code: a1.c`)

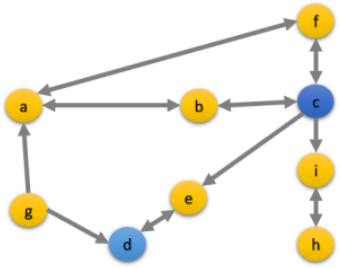
```
16      bool pathway[8] = {[0] = true, [2] = true};
```

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

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

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.

   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

1. Declare and initialize a road_networks multidimensional array that represents the adjacency matrix.

```
15      /*2D array for the road networks; 1 for open 0 for closed*/
16      bool road_networks[X][Y]={
17          {1, 1, 0, 0, 0, 1, 0, 0, 0},
18          {1, 1, 1, 0, 0, 0, 0, 0, 0},
19          {0, 1, 1, 0, 1, 1, 0, 0, 0},
20          {0, 0, 0, 1, 1, 0, 0, 0, 0},
21          {0, 0, 0, 1, 1, 0, 0, 0, 0},
22          {1, 0, 1, 0, 0, 1, 0, 0, 0},
23          {1, 0, 0, 1, 0, 0, 1, 0, 0},
24          {0, 0, 0, 0, 0, 1, 0, 1, 1},
25          {0, 0, 0, 0, 0, 0, 0, 0, 1}
26      };
```

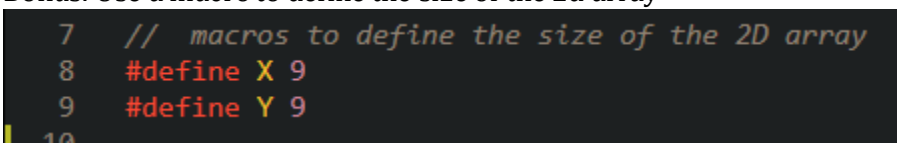
2. Display the adjacency matrix. Put a bracket to the points/destinations that are considered as charging stations, e.g. [c], [d]

```
C:\Users\Admin\Documents\BS COMSCI AKO\CMSC21\Lecture 6-7\b1.exe
    A   B   [C] [D] E   F   G   H   I
A   1   1   0   0   0   1   0   0   0
B   1   1   1   0   0   0   0   0   0
C   0   1   1   0   1   1   0   0   0
D   0   0   0   1   1   0   0   0   0
E   0   0   0   1   1   0   0   0   0
F   1   0   1   0   0   1   0   0   0
G   1   0   0   1   0   0   1   0   0
H   0   0   0   0   0   1   0   1   1
I   0   0   0   0   0   0   0   0   1
Which point are you located? 0-A, 1-B, 2-C, 3-D, 4-E, 5-F, 6-G, 7-H, 8-I
```

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. (Source code: b1.c)

```
C:\Users\Admin\Documents\BS COMSCI AKO\CMSC21\Lecture 6-7\b1.exe
    A   B   [C] [D] E   F   G   H   I
A   1   1   0   0   0   1   0   0   0
B   1   1   1   0   0   0   0   0   0
C   0   1   1   0   1   1   0   0   0
D   0   0   0   1   1   0   0   0   0
E   0   0   0   1   1   0   0   0   0
F   1   0   1   0   0   1   0   0   0
G   1   0   0   1   0   0   1   0   0
H   0   0   0   0   0   1   0   1   1
I   0   0   0   0   0   0   0   0   1
Which point are you located? 0-A, 1-B, 2-C, 3-D, 4-E, 5-F, 6-G, 7-H, 8-I
5
Current Point: F
Now at point A
Now at point B
Now at point C
point: C arrived at the charging station
Press any key to continue . . .
```

4. Bonus: Use a macro to define the size of the 2d array



```
7   //  macros to define the size of the 2D array
8   #define X 9
9   #define Y 9
10
```

Explanation:

My strategy for this problem is outlined in the following:

- Declare the multidimensional array and define its size using macro's
- Print the matrix by using two for loops nested to each other to work through a 2-dimensional array
- To put a bracket to the charging stations (C and D), I used for loops to iterate through a declared array of destination and, using control-statements (if, else if), determined which are the charging stations and put them inside a bracket
- To find the nearest station, a for-loop is used to iterate through each list (row) in the array using the X-coordinate (cur_point) determined by the user and the index as the coordinates. The bool is checked until the program reaches a charging station using the for-loop and control-statements

Insight:

Working through this problem provided me with a deeper understanding of matrix and how they behave and can be controlled with the use of for-loops. It is somehow appalling at first to witness first hand as to how reading through an array can escalate in complexity just from a 1D list to 2D array. I barely managed with 2D array, what more if it the dimensions are 3 or more?

Also, what interests me the most is how my problem-solving skills was engaged greatly in this problem. I thought through every detail and found solutions using the past lessons. Just how interesting (and depressing) can programming in C get?