Name: Briana Jade Adricula Date: May 11, 2022

1. Revise line 16
   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.

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

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

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

    for (int i = 0; i < NUM_PATHWAYS; i++){
        if (pathway[i]){
            printf("pathway[%d] is open \n", i);
        }else{
            printf("pathway[%d] is close \n", i);
        }
        }

    return 0;
}
```

```
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

Process returned 0 (0x0)   execution time : 0.047 s
Press any key to continue.
```

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

main.c | main.c | main.c | main.c

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

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

    bool pathway[8] = {{true},{false},{true}}; // if inner isn't fill the remaining elements are initialized to false or 0
    for (int i = 0; i < NUM_PATHWAYS; i++){
        if (pathway[i]){
            printf("pathway[%d] is open \n", i);
        }else{
            printf("pathway[%d] is close \n", i);
        }
        }

    return 0;
}
```

```
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

Process returned 0 (0x0)   execution time : 0.047 s
Press any key to continue.
```

2.

```c
# include <stdio.h>

int main(){
int point; // declared variable
  int road_networks[8][8] = // initialized array
  {
    {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("    | A |  B | [C] | [D] | E  | F |  G |  H |\n-------------------------------------------------\n");

  // prints out the table of arrays
  for(int i = 0; i < 9; ++i)
  {
     printf("%d   |",i);
     for(int j = 0; j < 8; ++j)
     {
        if(road_networks[i][j] > 9)
        {
          printf(" %d |", road_networks[i][j]);
        } else {
          printf(" %d  |", road_networks[i][j]);
        }
     }
     printf("\n-------------------------------------------------\n");
  }

  // Shows where point are located using switch statements
          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);
          switch(point){
          case 0:
              printf("At point: A\n");
              break;
          case 1:
              printf("At point: B\n");
              break;
          case 2:
              printf("point: C is a charging station\n");
              break;
          case 3:
              printf("point: D is a charging station\n");
              break;
          case 4:
              printf("At point: E\n");
              break;
          case 5:
              printf("At point: F\n");
              break;
          case 6:
              printf("At point: G\n");
              break;
          case 7:
              printf("At point: H\n");
              break;
          }
// shows nearest charging station
    if (point = 0 || 1 || 5)
        printf("point: C arrived to charging station");
    else
        if(point = 4 || 7|| 6)
```

```c
            }
// shows nearest charging station
    if (point = 0 || 1 || 5)
        printf("point: C arrived to charging station");
    else
        if(point = 4 || 7|| 6)
        printf("point: D arrived to charging station");


  return 0;
}
```

Some test Cases:

```
Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H
0
At point: A
point: C arrived to charging station
Process returned 0 (0x0)   execution time : 130.770 s
Press any key to continue.
```

```
--------------------------------------------------
Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H
1
At point: B
point: C arrived to charging station
Process returned 0 (0x0)   execution time : 1.683 s
Press any key to continue.
```

The number 2 problem is challenging for me and hard to understand since I don't know how adjacent matrix works and how should it be implemented with the code. I tried learning about dijkstra's algorithm to get the shortest path but I need more time to really understand and to be implemented in the code so what I did instead is that I do it manually. I printed out the array manually, make a switch statement and if statement to answer the questions to be inputted and outputted. So, it just shows the printed matrix and then answers the questions right but it does not belong to what the goal is about the adjacent matrix. But I think, it I had the time to learn about dijkstra's algorithm it could possibly work to get the shortest path.