Mary Nicolette Parcon
BSCS CMSC 21-1

1.

a.

```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 2 are open for transportation. The rest are close.

    */
    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;
}
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
PS C:\Users\mjpar\Documents\UP\SEM 2\CMSC 21-1 Lab\Lab Exercises> cd "c:\Users\mjpar\Documents\UP\SEM 2\CMSC 21-1 Lab\Lab Exercises\" ; i
f ($?) { gcc test.c -o test } ; if ($?) { .\test }
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
PS C:\Users\mjpar\Documents\UP\SEM 2\CMSC 21-1 Lab\Lab Exercises>
```

b.

```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 2 are open for transportation. The rest are close.

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

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
PS C:\Users\mjpar\Documents\UP\SEM 2\CMSC 21-1 Lab\Lab Exercises> cd "c:\Users\mjpar\Documents\UP\SEM 2\CMSC 21-1 Lab\Lab Exercises\" ; i
f ($?) { gcc test.c -o test } ; if ($?) { .\test }
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
PS C:\Users\mjpar\Documents\UP\SEM 2\CMSC 21-1 Lab\Lab Exercises>
```

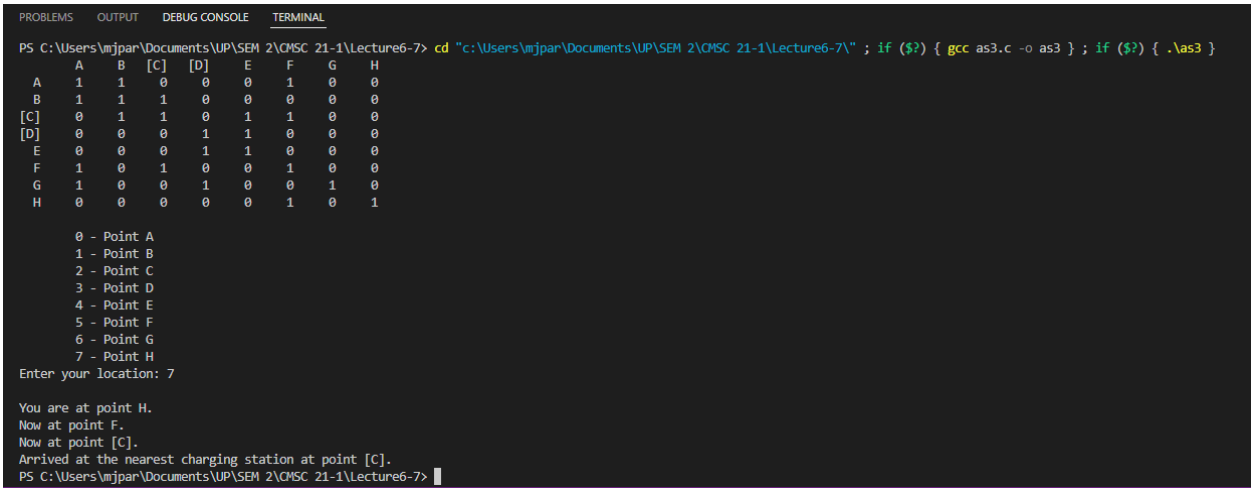
2.

```c
#include <stdio.h>

#define COL 8
#define ROW 8

int main(void) {

    char *letters[] = {"  A", "  B", "[C]", "[D]", "  E", "  F", "  G", "  H"};
    char *point[] = {"A", "B", "[C]", "[D]", "E", "F", "G", "H"};
    // initialize multidimensional array that represents the adjacency matrix
    int loc, road_networks[ROW][COL] = {{1,1,0,0,0,1,0,0}, // a
                                        {1,1,1,0,0,0,0,0}, // b
                                        {0,1,1,0,1,1,0,0}, // c
                                        {0,0,0,1,1,0,0,0}, // d
                                        {0,0,0,1,1,0,0,0}, // e
                                        {1,0,1,0,0,1,0,0}, // f
                                        {1,0,0,1,0,0,1,0}, // g
                                        {0,0,0,0,0,1,0,1}}; // h

    // col header
    printf("%6c", ' ');
    for (int i = 0; i < ROW; i++) {
        printf("%-6s", letters[i]);
    }
    printf("\n");

    // printing array in tabular form
    for (int i = 0; i < ROW; i++) {
        printf("%s", letters[i]); // row header

        for (int j = 0; j < COL; j++) {
            printf("%6i", road_networks[i][j]);
        }
        printf("\n");
    }

    printf("\n\t0 - Point A\n");
    printf("\t1 - Point B\n");
    printf("\t2 - Point C\n");
    printf("\t3 - Point D\n");
    printf("\t4 - Point E\n");
    printf("\t5 - Point F\n");
    printf("\t6 - Point G\n");
```

```
    printf("\t7 - Point H\n");

    while (1) { // while loop for input validation
        printf("Enter your location: ");
        scanf("%d", &loc);

        if (loc < 0 || loc > ROW - 1) { // checks if input is not 0 through 7
            printf("\nError! You can only be at points 1 through 7. Make sure you
type only the number.\n");
        }
        else { // if valid, break
            printf("\n");
            break;
        }
    }

    printf("You are at point %s.\n", point[loc]);

    for (int i = 0; i < COL; i++) {
        if (road_networks[loc][2] == 1) { // if loc directly at point C
            printf("Now at point %s.\n", point[2]);
            printf("Arrived at the nearest charging station at point %s.",
point[2]);
            break;
        }

        else if (road_networks[loc][3] == 1) { // if loc directly at point D
            printf("Now at point %s.\n", point[3]);
            printf("Arrived at the nearest charging station at point %s.",
point[3]);
            break;
        }

        else if (road_networks[loc][i] == 1) {
            printf("Now at point %s.\n", point[i]);

            if (point[i] == point[2] || point[i] == point[3]) {
                printf("Arrived at the nearest charging station at point %s.",
point[i]);
                break;
            }

            else { // reiterate loop if loc not at charging point
                loc = i;
            }
```

```
        }
    }
    return 0;
}
```

Output:



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS C:\Users\mjpar\Documents\UP\SEM 2\CMSC 21-1\Lecture6-7> cd "c:\Users\mjpar\Documents\UP\SEM 2\CMSC 21-1\Lecture6-7\" ; if ($?) { gcc as3.c -o as3 } ; if ($?) { .\as3 }
      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

      0 - Point A
      1 - Point B
      2 - Point C
      3 - Point D
      4 - Point E
      5 - Point F
      6 - Point G
      7 - Point H
Enter your location: 7

You are at point H.
Now at point F.
Now at point [C].
Arrived at the nearest charging station at point [C].
PS C:\Users\mjpar\Documents\UP\SEM 2\CMSC 21-1\Lecture6-7>
```