Loops and Arrays

Lecture 4 Assignments

1. What is the output of the following program?

```
#include <stdio.h>

int main(void)
{
 int i;

 i = 1;
 while (i <= 128) {
 printf("%d ", i);
 i *= 2;
 }

 return 0;
}
```

Save your code as as1.c

```
1   #include <stdio.h>
2
3   int main(void){
4
5       int i;
6       i = 1;
7       while(i <= 128){
8           printf("%d ", i);
9           i*= 2;
10      }
11
12      return 0;
13  }
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
[Running] cd "c:\Users\Ley\Downloads\CMSC21\mp
"c:\Users\Ley\Downloads\CMSC21\mp1\"as1
1 2 4 8 16 32 64 128
[Done] exited with code=0 in 0.654 seconds
```

This program outputs a sequence of powers of 2 starting from 1 until 128. This was possible because i was first initialized to 1 then the while loop had a condition that if "while" was less than or equal to 128, it would print the current value of i and update the i variable by multiplying it by 2.

2. Which one of the following statements is not equivalent to the other two (assuming that the loop bodies are the same)?

```
a) while (i < 10) {...}
b) for (; i < 10;) {...}
c) do {...} while (i < 10);
```

Save your code as as2.c

OUTPUT:

```
For loop:
-----

While loop:
-----

Do-while loop:
10
-----
```

```c
#include <stdio.h>

int main(void){
    int i = 10;

    // for loop
    printf("For loop: \n");
    for(;i<10;){
        printf("%d", i);
    }
    printf("----- \n\n");

    // while loop
    printf("While loop: \n");
    while(i<10){
        printf("%d", i);
    }
    printf("----- \n\n");

    // do-while loop
    printf("Do-while loop: \n");
    do{
        printf("%d", i);
    }while(i<10);

    printf(" \n----- \n\n");


    return 0;

}
```

The three loops are the same in terms of their conditions and logic. However, do-while loops execute the loop body at least once before checking its condition. For example, the code above has the same loop bodies for the three kinds of loops. "i" is not less than 10 so the for-loop and while loop did not print anything. The do-while loop on the other hand, printed "10". Thus the do-while loop is not equivalent with the other two loops.

3. Convert item 1 into an equivalent for statement. You can validate your answer by checking if the produced outputs by both the while and for statements are similar.

Save your code as as3.c

```c
#include <stdio.h>

int main(void){
    int i;

    for(i = 1; i<= 128; i*= 2){
        printf("%d ", i);
    }
    return 0;
}
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
[Running] cd "c:\Users\Ley\Downloads\CMSC21\mp1"
"c:\Users\Ley\Downloads\CMSC21\mp1\"as3
1 2 4 8 16 32 64 128
[Done] exited with code=0 in 0.255 seconds
```

4. Write a code that computes for the power of two:

TABLE OF POWERS OF TWO

n	2 to the n
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1024

Save your code as as4.c

```c
#include <stdio.h>

int main(void) {

    int n, power = 1 ; // declare variables n and power, and initialize power to 1

    // print table header
    printf("n   2 to the n \n");
    printf("--- ---------- \n");

    // loop from n=0 to n=10 and print power of 2 for each value of n
    for(n = 0; n <= 10; n++) {
        // print n and 2^n with appropriate formatting
        printf("%d %10d \n", n, power);
        // update power by doubling it
        power *= 2;
    }

    return 0;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL Code

```
n   2 to the n
--- ----------
0          1
1          2
2          4
3          8
4         16
5         32
6         64
7        128
8        256
9        512
10       1024
```

5. Write a program that displays a one-month calendar.

```
Enter number of days in month: 31
Enter the starting day of the week (1=Sun, 7=Sat): 3

       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```

There should be a user prompt to set:

- **The number of days**
- **The day of the week on which the month begins.**

Additionally, add checkers to validate whether the days entered are valid. For instance, the following number of days are invalid: 32, -1, 0, 27.

This addition will be a good refresher to our previous topic, selection statements. `Save your code as as5.c`

```c
#include <stdio.h>

int main(void){

    int days, week;

    printf("One-month Calendar \n");

    // Asks user for the number of days
    // If an input less than 28 or greater than 31 is entered, the programs asks the user again
    do{
        printf("Enter number of days in month: ");
        scanf("%d", &days);

    }while(days < 28 || days > 31 );

    // Asks user for the number of days
    // If an input less than 0 or greater than 7 is entered, the programs asks the user again

    do{
        printf("Enter the starting day of the week (1 = Sun, 7 = Sat): ");
        scanf("%d", &week);
    }while(week < 1 || week > 7);


    // print newlines before the spaces
    printf("\n\n");

    // subtract 1 from week to adjust for the first day of the month
    for( int i = 1; i<=week-1; i++){
        printf("   ");
    }

    // loop for printing the days of the month
    for( int i = 1; i <= days; i ++){
        printf("%3d", i);

        if((i + week - 1) % 7 == 0 ){
            printf("\n");
        }
    }

    return 0;
}
```

```
One-month Calendar
Enter number of days in month: 31
Enter the starting day of the week (1 = Sun, 7 = Sat): 3


       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```

6. 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.

```
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] = {true, false, true, false, false, false, false, false};
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 is 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 }
```

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.

- 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)

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

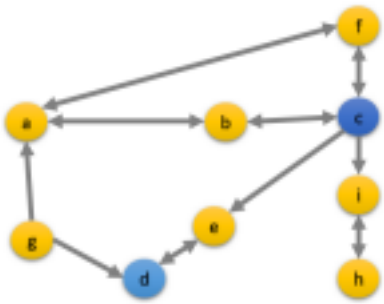
7. A road network can be represented using graphs. Assuming we have points / stations a, b, c, d, e, f, g, and h, we can represent a direct path from a point to another point using arrows.

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

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

#define NUM_VERTICES 8
#define ROAD_NETWORKS_SIZE NUM_VERTICES

#define CHARGING_STATION_C 'c'
#define CHARGING_STATION_D 'd'
```

```
/*  2D array that represents the road network.
 Each row and column represents a vertex, and the values indicate whether there
is a road between the vertices.
 The values in the third and fourth rows represent the charging stations.*/
int road_networks[ROAD_NETWORKS_SIZE][ROAD_NETWORKS_SIZE] = {
{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 [charging station]
{0, 0, 0, 1, 1, 0, 0, 0},   // d [charging station]
{0, 0, 1, 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
};



// array that maps the index of a vertex in the road network matrix to its
character representation.
char charging_stations[NUM_VERTICES] =
  { 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h' };


int
main ()
{

    // Display adjacency matrix
    printf ("Road Network:\n");

printf ("     %c   %c  [%c] [%c]  %c   %c   %c   %c\n",
       charging_stations[0], charging_stations[1], CHARGING_STATION_C,
       CHARGING_STATION_D, charging_stations[4], charging_stations[5],
       charging_stations[6], charging_stations[7]);

for (int i = 0; i < ROAD_NETWORKS_SIZE; i++)
```

```
    {

if (i == 2 || i == 3)
    {

printf ("[%c] ", charging_stations[i]); // print vertex character with brackets
on the left of 'c' and 'd'
    }
      else
    {

printf (" %c  ", charging_stations[i]); // print vertex character
    }

for (int j = 0; j < ROAD_NETWORKS_SIZE; j++)
    {

if ((j == 2 || j == 3) || (i == 2 || i == 3))
        {

printf ("[%d] ", road_networks[i][j]);  // print adjacency matrix value with
brackets
        }
      else
        {

printf (" %d  ", road_networks[i][j]);  // print adjacency matrix value
        }

}

printf ("\n");

}


printf ("\n");
```

```
// Prompt user for input
printf("Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6
- G, 7 - H\n\n");
int point_index;
scanf("%d", &point_index);


// Find nearest charging station for the selected point
char point = charging_stations[point_index];
int nearest_index = -1;
int shortest_distance = INT_MAX;

printf("At point: %c \n", point);

// iterate through each charging station and find the closest one to the current
point
for (int j = 0; j < ROAD_NETWORKS_SIZE; j++) {
    if ((charging_stations[j] == CHARGING_STATION_C || charging_stations[j] ==
CHARGING_STATION_D)) { // check if the current vertex is a charging station
        int distance = abs(j - point_index); // calculate distance between
charging station and current point
        if (distance < shortest_distance) {
            nearest_index = j;
            shortest_distance = distance;
        }
    }
}

// print result for selected point
if (nearest_index == -1) {
    printf("No nearest charging station found for point %c\n", point);
} else {
    char nearest = charging_stations[nearest_index];
    printf("point: %c arrived to charging station \n", nearest);
}
```

```
return 0;

}
```

Output:

```
Road Network:
     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  [1] [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

The nearest charging station to point a is c.
The nearest charging station to point b is c.
The nearest charging station to point c is c.
The nearest charging station to point d is d.
The nearest charging station to point e is d.
The nearest charging station to point f is d.
The nearest charging station to point g is d.
The nearest charging station to point h is d.
```

Output:

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

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

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

2
At point: c
point: c arrived to charging station

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

3
At point: d
point: d arrived to charging station

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

4
At point: e
point: d arrived to charging station

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
point: d arrived to charging station

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

6
At point: g
point: d arrived to charging station

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

7
At point: h
point: d arrived to charging station

Instructions for submissions

● Take screenshots of your codes for numbers which requires coding (e.g., 1, 2, 3) and embed it on the pdf along with an example output.

● Submit your answers in a pdf file with filename assignment2[surname].pdf ● Save the pdf file (assignment4[surname].pdf) and the codes in the directory: CMSC21/Lecture4/Assignments/

● Remember that you have initially created this repository for your reading assignment. ● Upload to github.

● Download git cmd

● Navigate to the CMSC21 Folder

● For example (assuming your CMSC21 folder is in Documents)

- **`cd Documents/CMSC21`**
- **`git add –all`**
- **`git commit -m "Lecture 4 Assignment"`**
- **`git push -u origin main T`**

▪ Email me the github link with subject: [CMSC 21] Assignment 4 (Surname) [Date Submitted]