RIAN LUIS C. MABAIT

BS in Computer Science – I

1.

```c
#include <stdio.h>

int main(){
    int i = 1;

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



    }
    return 0;

    // prints 1, 2, 4, 8, 16, 32, 64, 128
}
```

SAMPLE OUTPUT:

```
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments> cd "c:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments\" ; if ($?) { gcc as1.c -o as1 } ; if ($?) { .\as1 }
1 2 4 8 16 32 64 128
```

2.

```c
#include <stdio.h>

void task1(){
    printf("\n");

    int i = 1;
    while (i < 10){
        printf("%d ", i);
        i++;
    }
    printf("\n");
}
void task2(){
    int  i = 1;
    for (; i < 10; i++){
        printf("%d ", i);
    }
    printf("\n");
}
void task3(){
    int i = 1;
    do{
        printf("%d ", i);
        i++;
    }while (i<10);
    printf("\n");


}

int main(){
    task1();
    task2();
    task3();
}
```

SAMPLE OUTPUT:

```
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments> cd "c
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

Although the outputs are the same, the do while statement is not equivalent to the other 2 because it checks the condition at the end of the iteration while the for loop and the while loop checks the condition first before iterating.

3.

```c
#include <stdio.h>

int main(){

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

    return 0;
}
```

SAMPLE OUTPUT:

```
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments> cd
1 2 4 8 16 32 64 128
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

4.

Lecture4 > Assignments > C as4.c > ⊕ main()

```c
#include <stdio.h>

int main(){

    int num, count, i; // declares as type int

    // asks for user input
    printf("Enter an integer n: ");
    scanf("%d", &num);

    // prints table format
    printf("\nTABLE OF POWERS OF 2\n");
    printf("n          2 to the n \n");
    printf("_ _ _      _ _ _ _ _ _\n\n");

    // sets count into 1
    count = 1;

    // for loop that calculates 2 to the nth power and prints in a tabular way.
    for (i = 0; i<=num; i++){
        printf("%3d          %3d\n", i, count);
        count *=2;
    }
    return 0;
}
```

SAMPLE OUTPUT:

```
PS C:\Users\RIAN\Desktop\CMSC21> cd "c:\Users\RIAN\Desk
Enter an integer n: 5

TABLE OF POWERS OF 2
n          2 to the n
_ _ _      _ _ _ _ _ _

  0          1
  1          2
  2          4
  3          8
  4          16
  5          32
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

```
Enter an integer n: 0

TABLE OF POWERS OF 2
n          2 to the n

_ _ _      _ _ _ _ _ _

  0          1
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

5.

```c
#include <stdio.h>

void calendar(int no_of_days, int day_of_week){

    printf("\n");
    printf("  S   M   T   W  TH   F  SA\n");
    int i;

    // prints the spaces
    for (i = 1; i < day_of_week; i++){
        printf("    ");
    }


    // prints the calendar
    for (i = 1; i <= no_of_days; i++){
        printf("%3d ", i);

        // prints new line if it's divisible by the number of days in a week
        if((i + day_of_week - 1) % 7 == 0){
            printf("\n");
        }
    }
}
```

```c
int main(){

    int no_of_days, day_of_week; // variable declaration

    // asks for number of days in month
    printf("\nEnter number of days in the month: ");
    scanf("%d", &no_of_days);

    // checks for errors
    switch (no_of_days){

    case 30: case 31: case 28: case 29:

        // asks for starting day of the week
        printf("\nEnter starting day of the week (1 = Sun, 7 = Sat): ");
        scanf("%d", &day_of_week);

        switch (day_of_week){
        case 1: case 2: case 3: case 4: case 5: case 6: case 7:
            calendar(no_of_days, day_of_week); // function call
            break;

        default:
            printf("Invalid starting day of the week!");
            main(); // will call the main function if user inputted something out of the choice spectrum
            break;
        }

        break;

    default:
        printf("\nInvalid no. of days entered!");
        main(); // will call the main function if user inputted something out of the choice spectrum
        break;
    }

}
```

SAMPLE OUTPUT:

```
Enter number of days in the month: 31

Enter starting day of the week (1 = Sun, 7 = Sat): 3

 S   M   T   W  TH   F  SA
             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
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

31 days in a month

```
Enter number of days in the month: 0

Invalid no. of days entered!
Enter number of days in the month: 31

Enter starting day of the week (1 = Sun, 7 = Sat): 0
Invalid starting day of the week!
Enter number of days in the month:
```

In the case that the user input invalid options, it will ask the user again for valid input

```
Enter number of days in the month: 29

Enter starting day of the week (1 = Sun, 7 = Sat): 7

 S   M   T   W  TH   F  SA
                         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
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

29 days in a month

```
Enter number of days in the month: 30

Enter starting day of the week (1 = Sun, 7 = Sat): 1

 S   M   T   W  TH   F  SA
 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
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

30 days in a month

```
Enter number of days in the month: 28

Enter starting day of the week (1 = Sun, 7 = Sat): 2

 S   M   T   W  TH   F  SA
     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
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

28 days in a month

6.

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

```c
1   #include <stdio.h>
2   #include <stdbool.h>
3   #define NUM_PATHWAYS ((int) (sizeof(pathway) / sizeof(pathway[0])))
4
5   int main(){
6       // a.
7       bool pathway[8] = { [0]=true, [2]=true }; // designated initializers
8
9       for (int i = 0; i < NUM_PATHWAYS; i++){
10          if (pathway[i]){
11              printf("pathway[%d] is open \n", i);
12          }
13
14          else{
15              printf("pathway[%d] is close \n", i);
16          }
17      }
18
19      return 0;
20
21  }
```

SAMPLE OUTPUT:

```
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
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
```

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

```c
Lecture4 > Assignments > C as6_b.c > ⊕ main()
1   #include <stdio.h>
2   #include <stdbool.h>
3   #define NUM_PATHWAYS ((int) (sizeof(pathway) / sizeof(pathway[0])))
4
5   int main(){
6       // b.
7       bool pathway[8] = {true,false,true}; // without designated initializer
8
9       for (int i = 0; i < NUM_PATHWAYS; i++){
10          if (pathway[i]){
11              printf("pathway[%d] is open \n", i);
12          }
13
14          else
15              printf("pathway[%d] is close \n", i);
16
17      }
18
19      return 0;
20
21  }
```

SAMPLE OUTPUT:

```
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments> cd
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\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

7.

```c
#include <stdio.h>
#define ROWS 9
#define COLUMNS 9

int main(){

    int road_networks[ROWS][COLUMNS] = {
        {1,1,0,0,0,1,0,0,0},
        {1,1,1,0,0,0,0,0,0},
        {0,1,1,0,1,1,0,0,0},
        {0,0,0,1,1,0,0,0,0},
        {0,0,0,1,1,0,0,0,0},
        {1,0,1,0,0,1,0,0,0},
        {1,0,0,1,0,0,1,0,0},
        {0,0,0,0,0,0,0,1,1},
        {0,0,0,0,0,0,0,1,1}

    };

    printf("\n================= THE ADJACENCY MATRIX =================\n\n");
    printf("    a    b    c    d    e    f    g    h    i\n\n");


    int i = 0;
    int j = 0;

    // initializes the adjancy matrix with brackets on [c] & [d].
    while (i < ROWS){
        printf("%c ", 'a'+i);

        // checks if the iteration is at row and col 2 and 3. Also prints the adjancy matrix with brackets on [c] & [d].
        while (j < COLUMNS){
            i == 2 || j == 2 || i == 3 || j == 3 ? printf("[%d] \t", road_networks[i][j])
                : printf("%2d \t", road_networks[i][j]);
            j++;
        }
        i++;
        j = 0;
        printf("\n");
    }

    int point;
    printf("\n============= APPROACH NO. 1 =============\n");

    // user input
    printf("\nWhich point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I: \n");
    scanf("%d", &point);

    // aproach 1 by using a switch statement to know the closest charging station when looking at the graph.
    switch (point){
    case 0:
        printf("\nAt point: A\npoint: C is the nearest charging station\n");
        break;
    case 1:
        printf("\nAt point: B\npoint: C is the nearest charging station\n");
        break;
    case 2:
        printf("\nAt point: C\npoint: C is a charging station\n");
        break;
    case 3:
        printf("\nAt point: D\npoint: D is a charging station\n");
        break;
    case 4:
        printf("\nAt point: E\npoint: D is the nearest charging station\n");
        break;
    case 5:
        printf("\nAt point: F\npoint: C is the nearest charging station\n");
        break;
    case 6:
        printf("\nAt point: G\npoint: D is the nearest charging station\n");
        break;
    case 7:
        printf("\nAt point: H\nNo charging stations nearby\n");
        break;

    case 8:
        printf("\nAt point: I\nNo charging stations nearby\n");
        break;

    default:
        printf("\nMake sure to choose from the choices!\n");
        main(); // asks the user again to input a valid choice.
        break;
```

```c
86    // approach 2, using a for loop
87
88    // user input
89    printf("\n============= APPROACH NO. 2 =============\n");
90
91    printf("\nWhich point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I: \n");
92    scanf("%d", &point);
93
94    printf("\nAt point: %c\n", 'A' + point); // prints the point
95
96    // checks for errors
97    if (point > 8 || point < 0){
98        printf("\nPlease choose from the choices only!\n");
99        main();
100   }
101
102   for (int i = 0; i < ROWS; i++) {
103
104       if (point == 2){
105           printf("\nC is a charging station."); // if point is at the index of c which is 2, prints that the user has arrived on a charging station
106           break;
107
108       } else if (point ==3){
109           printf("\nD is a charging station"); // if point is at the index of d which is 3, prints that the user has arrived on a charging station
110           break;
111
112       }
113       else if (point == 8){
114           printf("\nH has no nearby charging stations"); // if point is at index of i which is 8, prints that i has no nearby charging stations
115           break;
116
117       }
118       // checks if user input is not at index 2,3 or 8.
119       else if (i == point){
120
121           for (int j = i; j <= COLUMNS; j++){
122
123           // checks if there is a one way path between a point and c
124               if (road_networks[j][2] == 1){
125                   printf("\npoint: C is the nearest charging station");
126                   break;
127
128               }
129               // checks if there is a one way path between a point and d
130               else if (road_networks[j][3] == 1){
131                   printf("\npoint: D is the nearest charging station");
132                   break;
133
134               }
135               else {
136                   continue;
137
138               }
139           }
140       }
141   }
142
143   return 0;
144
145
146   }
147
```

SAMPLE OUTPUTS:

CASE OF VALID INPUTS

```
================= THE ADJACENCY MATRIX =================

      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     0     0     1     1
i    0     0    [0]   [0]    0     0     0     1     1

============= APPROACH NO. 1 =============

Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I:
0

At point: A
point: C is the nearest charging station

============= APPROACH NO. 2 =============

Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I:
0

At point: A

point: C is the nearest charging station
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

```
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments> cd "c:\Users\RIAN\Desktop\CMSC21\Lecture4

================= THE ADJACENCY MATRIX =================

    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    0    0    1    1
i   0    0   [0]  [0]   0    0    0    1    1

============= APPROACH NO. 1 =============

Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I:
8

At point: I
No charging stations nearby

============= APPROACH NO. 2 =============

Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I:
8

At point: I

H has no nearby charging stations
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

```
================= THE ADJACENCY MATRIX =================

    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    0    0    1    1
i   0    0   [0]  [0]   0    0    0    1    1

============= APPROACH NO. 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

At point: D
point: D is a charging station

============= APPROACH NO. 2 =============

Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I:
3

At point: D

point: D is a charging station
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

```
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments> cd "c:\Users\RIAN\Desktop\CMSC21\Lecture4\
================= THE ADJACENCY MATRIX =================

    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    0    0    1    1
i   0    0   [0]  [0]   0    0    0    1    1

============= APPROACH NO. 1 =============
Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I:
1

At point: B
point: C is the nearest charging station

============= APPROACH NO. 2 =============

Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I:
1

At point: B

point: C is the nearest charging station
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments>
```

CASE OF INVALID INPUT (Will ask the user to go on again from the top)

```
PS C:\Users\RIAN\Desktop\CMSC21\Lecture4\Assignments> cd "c:\Users\RIAN\Desktop\CMSC21\Lecture4\Assign

================ THE ADJACENCY MATRIX ================

    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    0    0    1    1
i   0    0   [0]  [0]   0    0    0    1    1

============= APPROACH NO. 1 =============

Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I:
9

Make sure to choose from the choices!

================ THE ADJACENCY MATRIX ================

    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    0    0    1    1
i   0    0   [0]  [0]   0    0    0    1    1

============= APPROACH NO. 1 =============

Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I:
0

At point: A
point: C is the nearest charging station

============= APPROACH NO. 2 =============

Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I:
9

At point: J

Please choose from the choices only!

================ THE ADJACENCY MATRIX ================

    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    0    0    1    1
i   0    0   [0]  [0]   0    0    0    1    1

============= APPROACH NO. 1 =============

Which point are you located? 0 - A, 1 - B, 2 - C, 3 - D, 4 - E, 5 - F, 6 - G, 7 - H, 8 - I:
|
```