

1.

```
1  #include <stdio.h>
2
3  int main(){
4      int i = 1;
5
6      while (i<=128){
7          printf("%d ", i);
8          i*=2;
9
10     }
11
12     return 0;
13
14     // prints 1, 2, 4, 8, 16, 32, 64, 128
15 }
```

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.

```
Lecture4 > Assignments > C as2.c > ...
1  #include <stdio.h>
2
3  void task1(){
4      printf("\n");
5
6      int i = 1;
7      while (i < 10){
8          printf("%d ", i);
9          i++;
10     }
11     printf("\n");
12 }
13 void task2(){
14     int i = 1;
15     for (; i < 10; i++){
16         printf("%d ", i);
17     }
18     printf("\n");
19 }
20 void task3(){
21     int i = 1;
22     do{
23         printf("%d ", i);
24         i++;
25     }while (i<10);
26     printf("\n");
27 }
28
29
30 int main(){
31     task1();
32     task2();
33     task3();
34 }
```

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.

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

SAMPLE OUTPUT:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

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()
1  #include <stdio.h>
2
3  int main(){
4      int num, count, i; // declares as type int
5
6      // asks for user input
7      printf("Enter an integer n: ");
8      scanf("%d", &num);
9
10
11      // prints table format
12      printf("\nTABLE OF POWERS OF 2\n");
13      printf("n      2 to the n \n");
14      printf(" _ _ _ _ _ \n\n");
15
16      // sets count into 1
17      count = 1;
18
19      // for loop that calculates 2 to the nth power and prints in a tabular way.
20      for (i = 0; i<=num; i++){
21          printf("%3d      %3d\n", i, count);
22          count *=2;
23      }
24      return 0;
25
26 }
```

SAMPLE OUTPUT:

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

```
#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");
        }
    }
}
```

```
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 };`

```
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};`

```
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.

```
Lecture4 > Assignments > C as7.c > main()
1  #include <stdio.h>
2  #define ROWS 9
3  #define COLUMNS 9
4  #define length_of_road_networks ((int) (sizeof(road_networks) / sizeof(road_networks[0]))) // bonus
5
6  int main(){
7
8      int road_networks[ROWS][COLUMNS] = {
9          {1,1,0,0,0,1,0,0,0},
10         {1,1,1,0,0,0,0,0,0},
11         {0,1,1,0,1,1,0,0,0},
12         {0,0,0,1,1,0,0,0,0},
13         {0,0,0,1,1,0,0,0,0},
14         {1,0,1,0,0,1,0,0,0},
15         {1,0,0,1,0,0,1,0,0},
16         {0,0,0,0,0,0,0,1,1},
17         {0,0,0,0,0,0,0,1,1}
18     };
19
20
21     printf("\n===== THE ADJACENCY MATRIX =====\n\n");
22     printf("   a   b   c   d   e   f   g   h   i\n\n");
23
24
25     int i = 0;
26     int j = 0;
27
28     // initializes the adjancy matrix with brackets on [c] & [d].
29     while (i < length_of_road_networks){
30         printf("%c ", 'a'+i);
31
32         // checks if the iteration is at row and col 2 and 3. Also prints the adjancy matrix with brackets on [c] & [d].
33         while (j < length_of_road_networks){
34             i == 2 || j == 2 || i == 3 || j == 3 ? printf("[%d] \t", road_networks[i][j])
35             : printf("%2d \t", road_networks[i][j]);
36             j++;
37         }
38         i++;
39         j = 0;
40         printf("\n");
41     }
42 }
```

```
43     int point;
44     printf("\n===== APPROACH NO. 1 =====\n");
45
46     // user input
47     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");
48     scanf("%d", &point);
49
50     // approach 1 by using a switch statement to know the closest charging station when looking at the graph.
51     switch (point){
52     case 0:
53         printf("\nAt point: A\npoint: C is the nearest charging station\n");
54         break;
55     case 1:
56         printf("\nAt point: B\npoint: C is the nearest charging station\n");
57         break;
58     case 2:
59         printf("\nAt point: C\npoint: C is a charging station\n");
60         break;
61     case 3:
62         printf("\nAt point: D\npoint: D is a charging station\n");
63         break;
64     case 4:
65         printf("\nAt point: E\npoint: D is the nearest charging station\n");
66         break;
67     case 5:
68         printf("\nAt point: F\npoint: C is the nearest charging station\n");
69         break;
70     case 6:
71         printf("\nAt point: G\npoint: D is the nearest charging station\n");
72         break;
73     case 7:
74         printf("\nAt point: H\nNo charging stations nearby\n");
75         break;
76
77     case 8:
78         printf("\nAt point: I\nNo charging stations nearby\n");
79         break;
80
81     default:
82         printf("\nMake sure to choose from the choices!\n");
83         main(); // asks the user again to input a valid choice.
84         break;
85     }
```

```

87 // approach 2, using a for loop
88
89 // user input
90 printf("\n===== APPROACH NO. 2 =====\n");
91
92 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");
93 scanf("%d", &point);
94
95 printf("\nAt point: %c\n", 'A' + point); // prints the point
96
97 // checks for errors
98 if (point > 8 || point < 0){
99     printf("\nPlease choose from the choices only!\n");
100     main();
101 }
102
103 for (int i = 0; i < length_of_road_networks; i++) {
104
105     if (point == 2){
106         printf("\npoint: C 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
107         break;
108     }
109     else if (point == 3){
110         printf("\npoint: D 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
111         break;
112     }
113
114     else if (point == 7){
115         printf("\nH has no nearby charging stations"); // if point is at index of h which is 7, prints that h has no nearby charging stations
116         break;
117     }
118
119     else if (point == 8){
120         printf("\nI has no nearby charging stations"); // if point is at index of i which is 8, prints that i has no nearby charging stations
121         break;
122     }
123
124     // checks if user input is not at index 2,3 or 8.
125     else if (i == point){
126
127         for (int j = 1; j <= length_of_road_networks; j++){
128
129             // checks if there is a one way path between a point and c
130             if (road_networks[j][2] == 1){
131                 printf("\npoint: C is the nearest charging station");
132                 break;
133             }
134
135             // checks if there is a one way path between a point and d
136             else if (road_networks[j][3] == 1){
137                 printf("\npoint: D is the nearest charging station");
138                 break;
139             }
140             else {
141                 continue;
142             }
143         }
144     }
145 }
146 }
147
148 return 0;
149 }
150
151

```

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
```

```
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]      [0]      [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\QMSC21\Lecture4\Assignments> cd "c:\Users\RIAN\Desktop\QMSC21\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:
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
```