

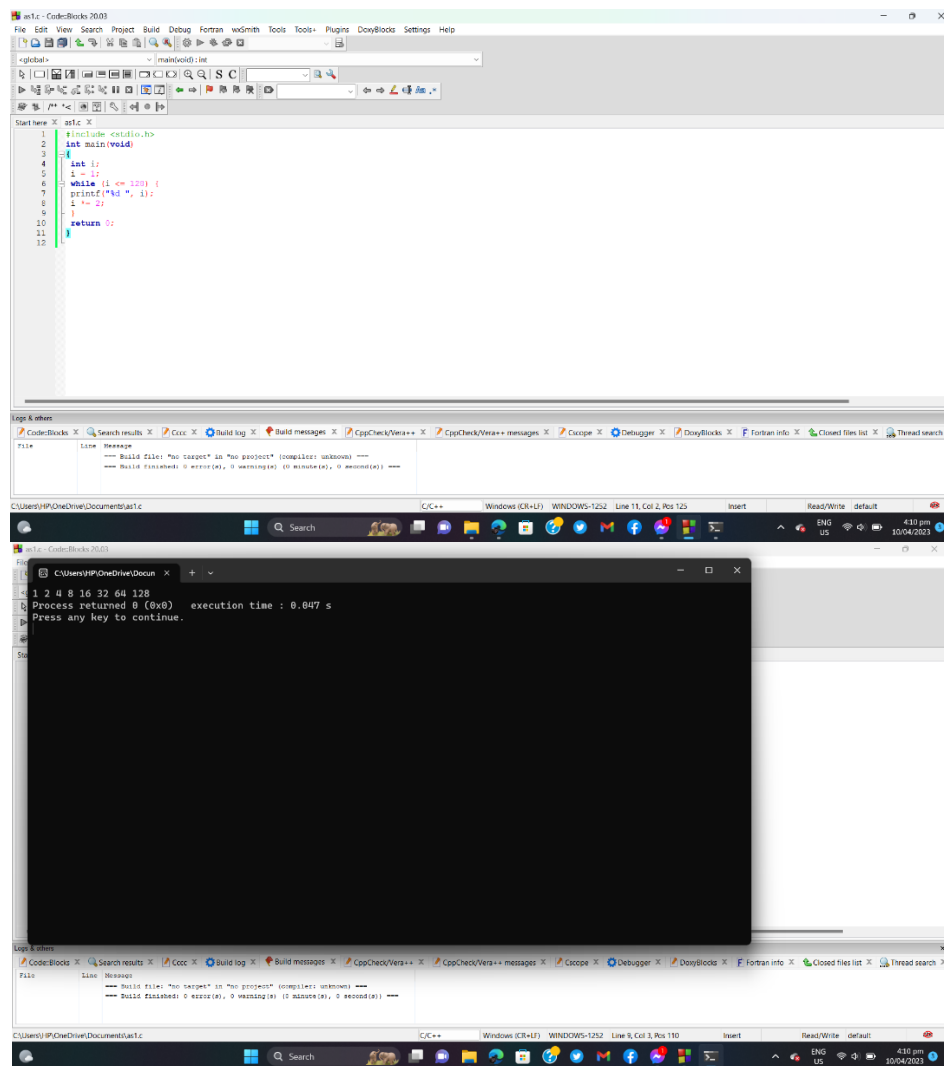
## Loops and Arrays

### 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;
}
```



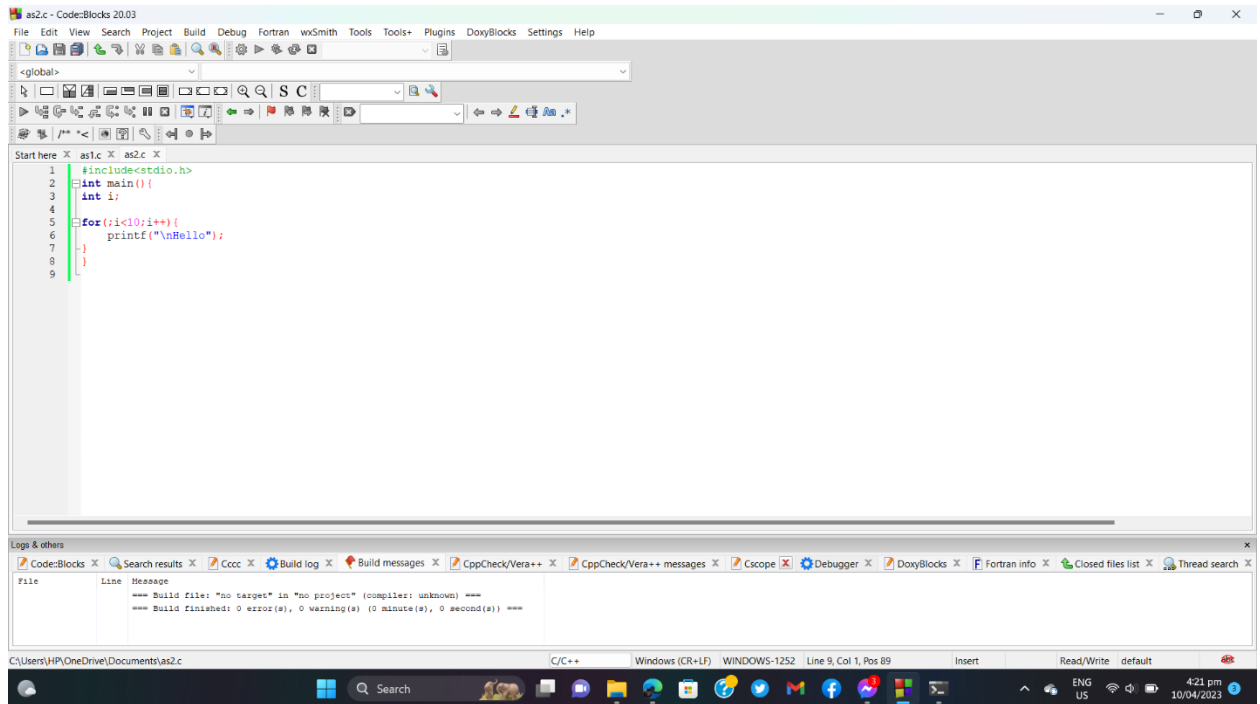
The screenshot shows the Code::Blocks IDE with a C program that prints powers of 2. The program is compiled and executed, and the output is displayed in a terminal window. The output shows the sequence of numbers: 1 2 4 8 16 32 64 128, followed by the message "Process returned 0 (0x0) execution time : 0.047 s" and "Press any key to continue." The IDE interface includes a menu bar, toolbar, and a status bar at the bottom.

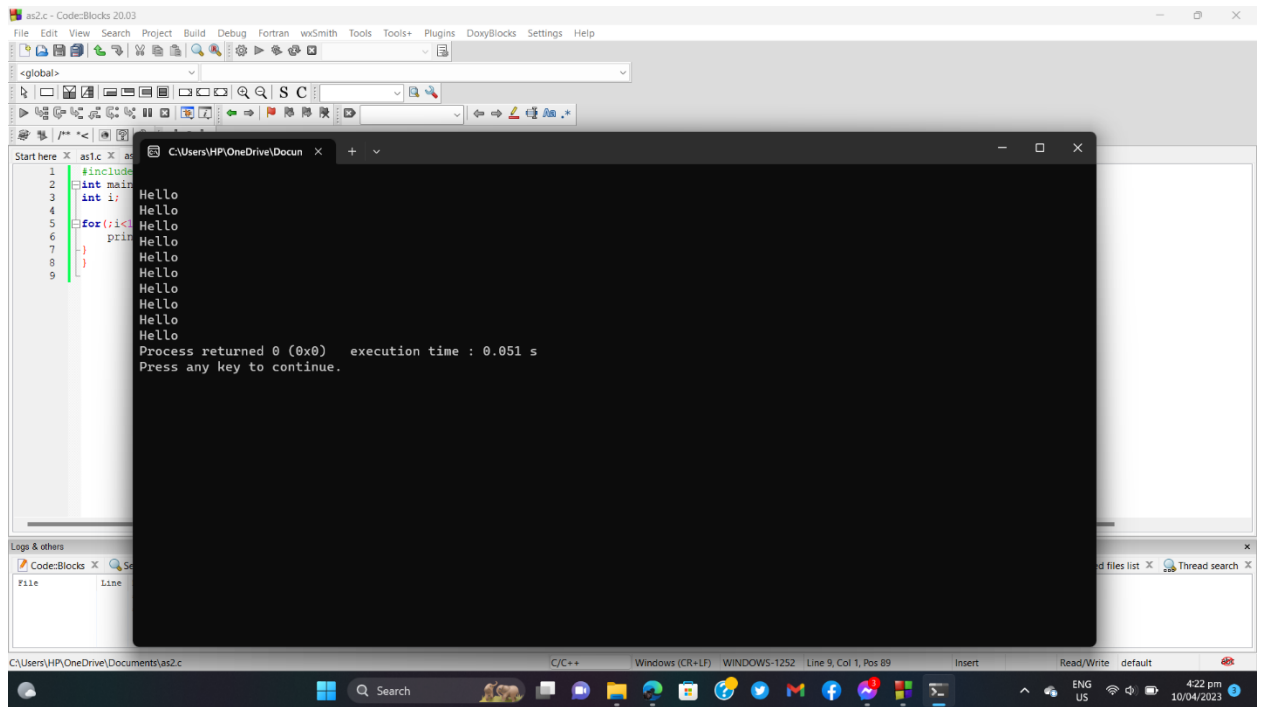
Save your code as as1.c

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);

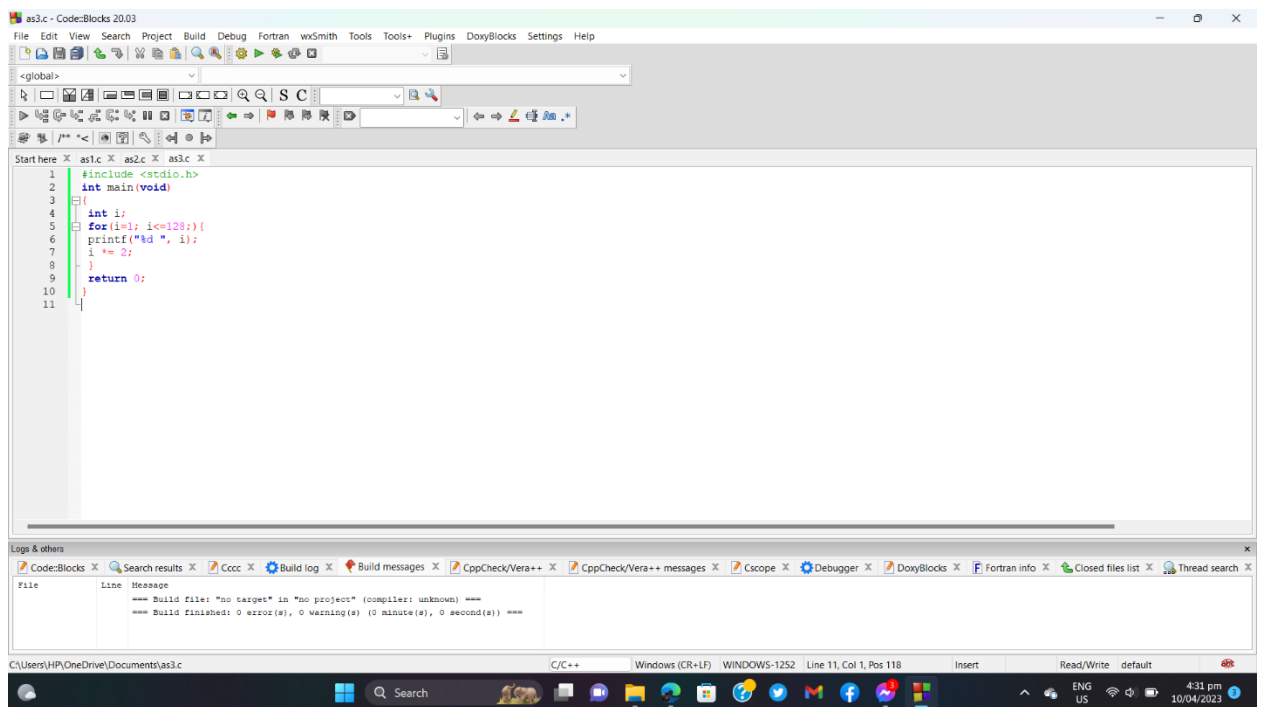
**Among the three letters b is the different since implementing a and c will result in infinity, however, on letter b it will run 10 times only.**

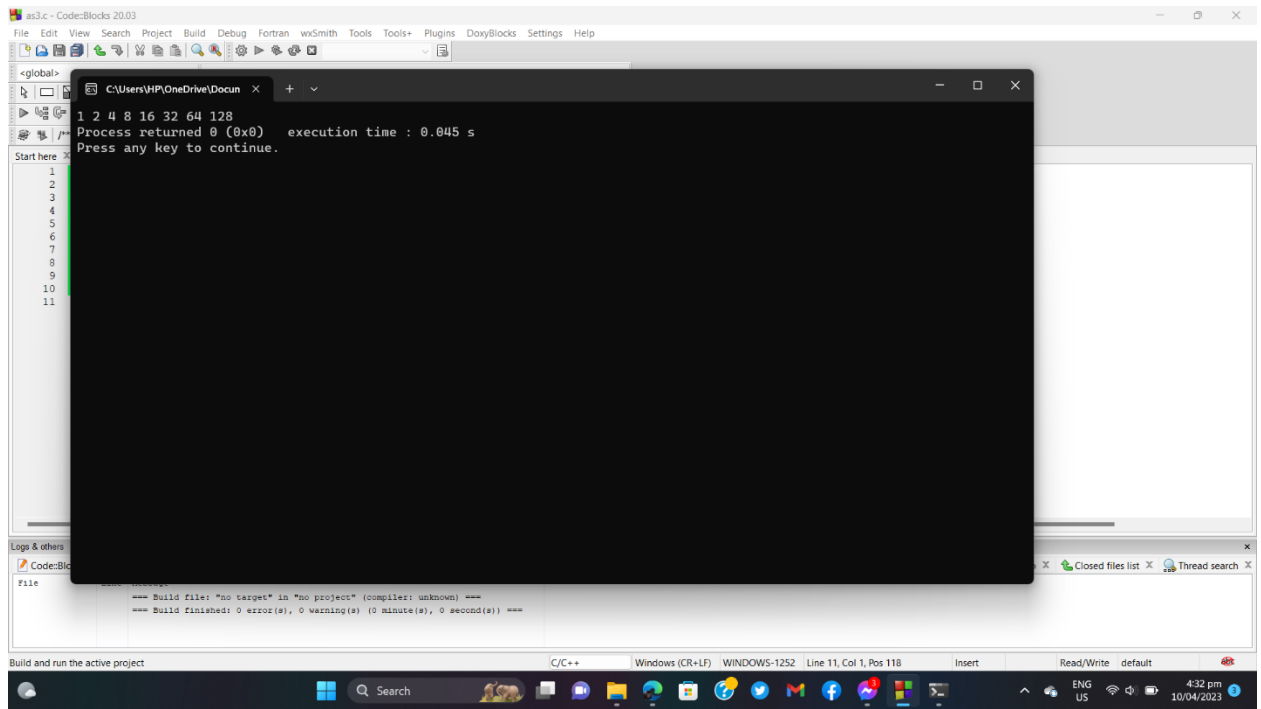




Save your code as `as2.c`

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.

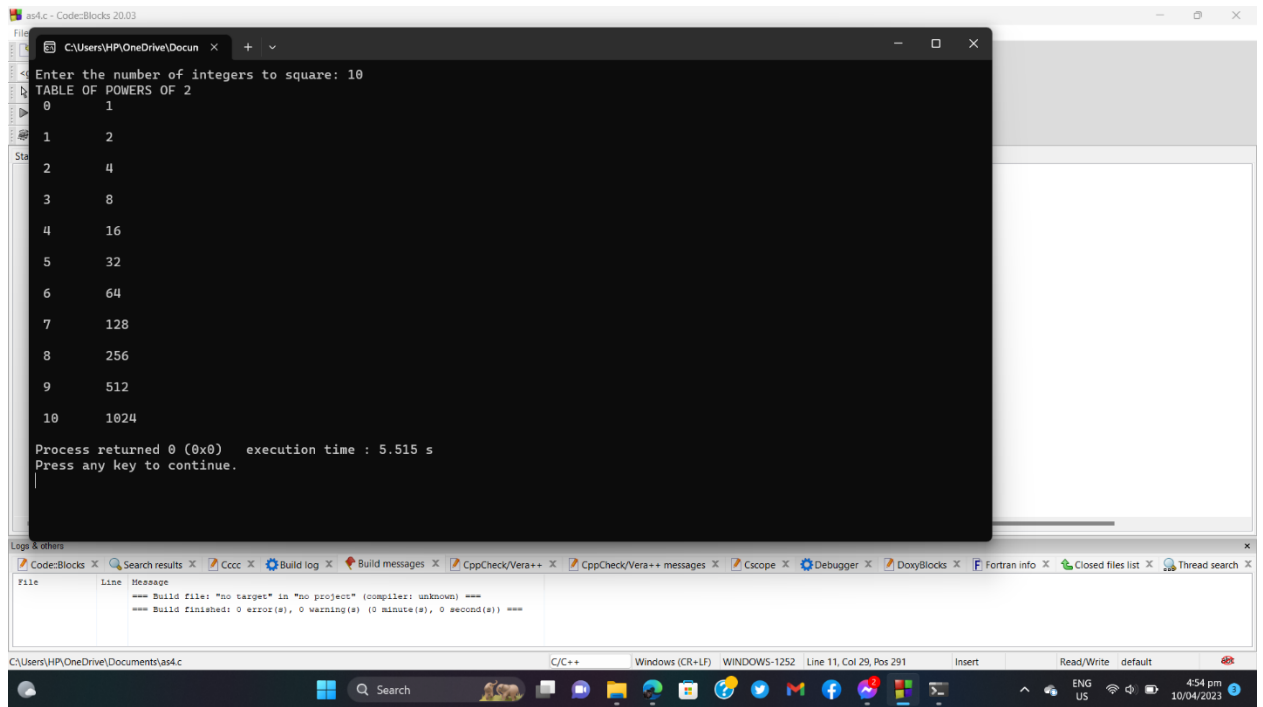
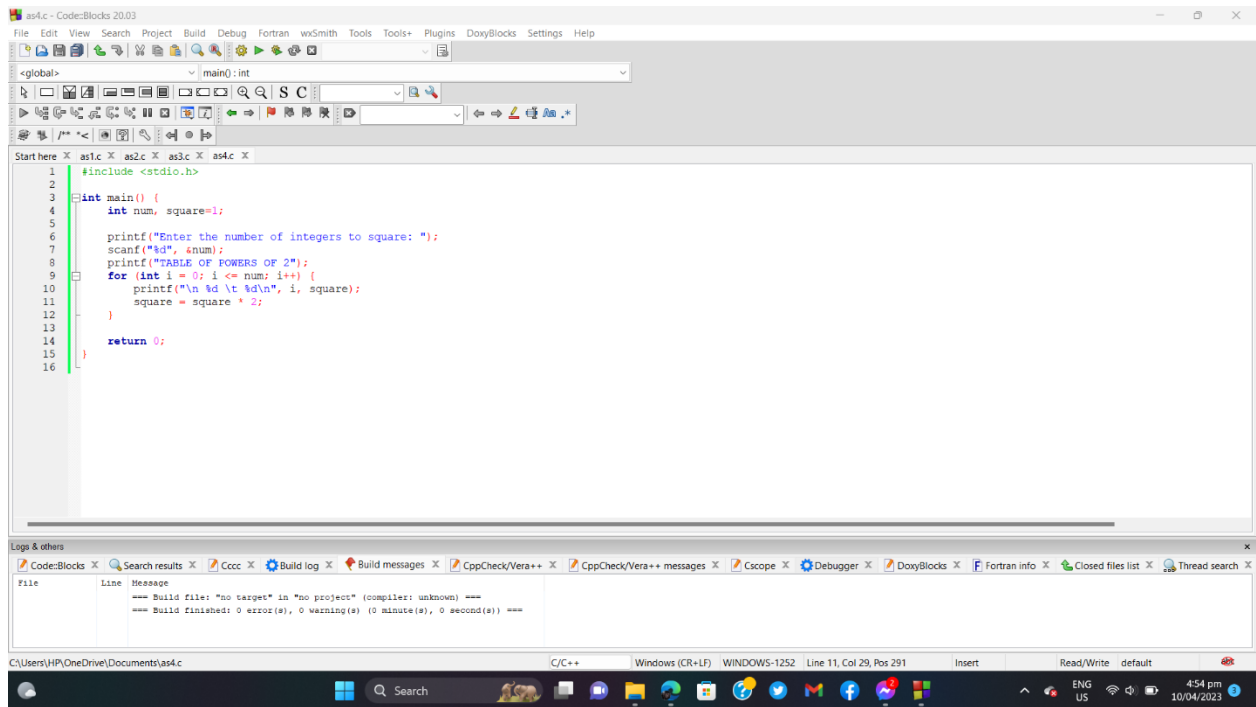




Save your code as as3.c

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

- 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

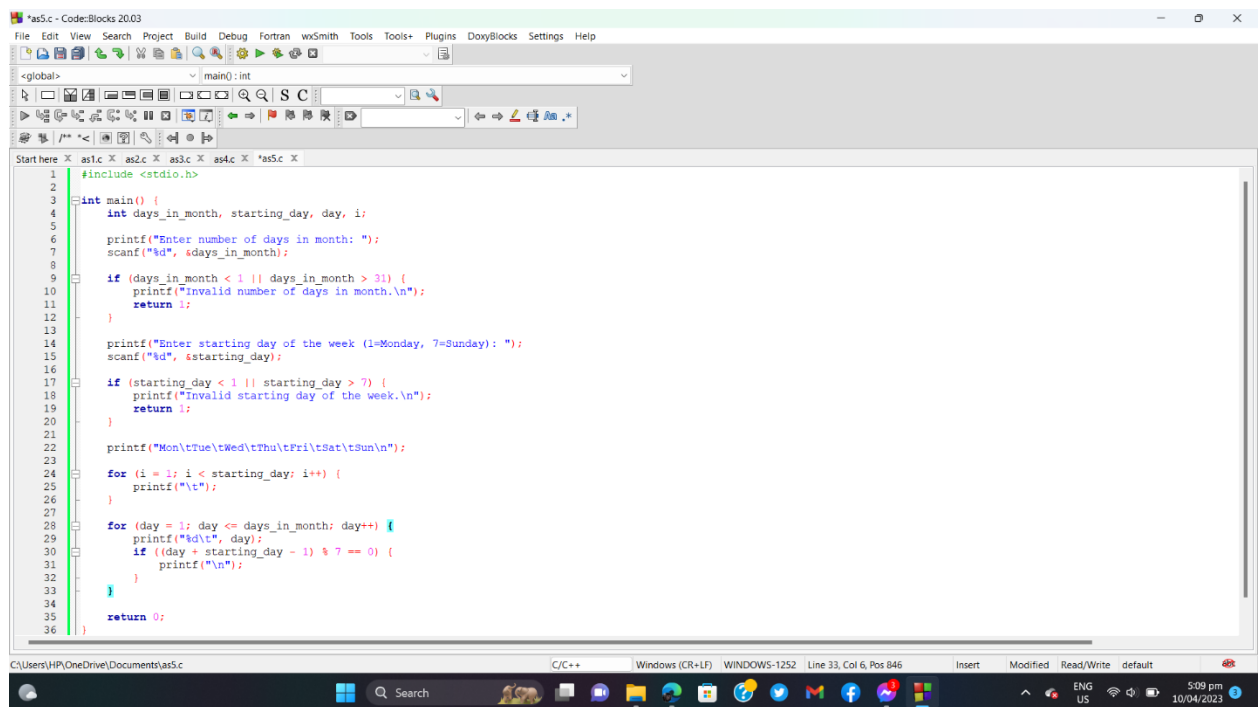
```

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.



```

1  #include <stdio.h>
2
3  int main() {
4      int days_in_month, starting_day, day, i;
5
6      printf("Enter number of days in month: ");
7      scanf("%d", &days_in_month);
8
9      if (days_in_month < 1 || days_in_month > 31) {
10         printf("Invalid number of days in month.\n");
11         return 1;
12     }
13
14     printf("Enter starting day of the week (1=Monday, 7=Sunday): ");
15     scanf("%d", &starting_day);
16
17     if (starting_day < 1 || starting_day > 7) {
18         printf("Invalid starting day of the week.\n");
19         return 1;
20     }
21
22     printf("Mon\tTue\tWed\tThu\tFri\tSat\tSun\n");
23
24     for (i = 1; i < starting_day; i++) {
25         printf("\t");
26     }
27
28     for (day = 1; day <= days_in_month; day++) {
29         printf("%d\t", day);
30         if ((day + starting_day - 1) % 7 == 0) {
31             printf("\n");
32         }
33     }
34
35     return 0;
36 }

```

```
as5.c - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global>
C:\Users\HP\OneDrive\Docum x + v
Enter number of days in month: 30
Enter starting day of the week (1=Monday, 7=Sunday): 3
Mon Tue Wed Thu Fri Sat Sun
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
Process returned 0 (0x0) execution time : 3.844 s
Press any key to continue.
return 0;
```

Save your code as as5.c

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.

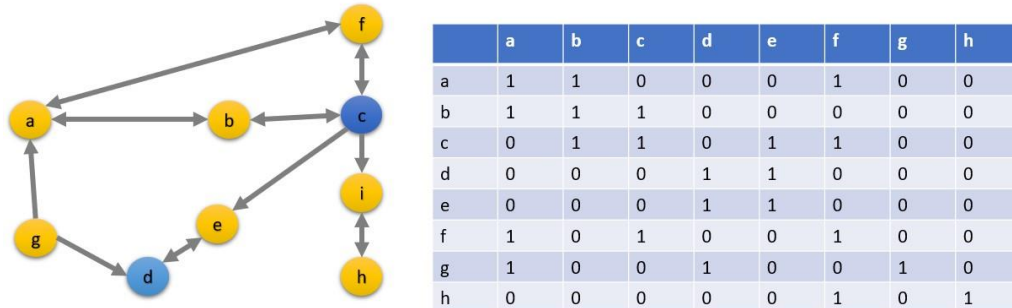
For example, based on the graph below:

- 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,  $a \rightarrow b = 1$  and  $b \rightarrow a = 1$  given that there's a two-way direct path between a and b. Meanwhile,  $a \rightarrow c = 0$  since there is no direct path between a and c. Moreover,  $a \rightarrow g = 0$  but  $g \rightarrow a = 1$  since there is a one-way path from point g to point a.



As a programming assignment:

1. Declare and initialize a `road_networks` multidimensional array that represents the adjacency matrix

```
int road_networks[NUM_POINTS][NUM_POINTS] = {
    {0, 1, 0, 0, 0, 1, 0, 0},
    {1, 0, 0, 0, 0, 1, 0, 0},
    {0, 0, 0, 0, 0, 1, 0, 0},
    {0, 0, 0, 0, 1, 0, 0, 0},
    {0, 0, 0, 1, 0, 0, 0, 0},
    {1, 1, 1, 0, 0, 0, 1, 0},
    {0, 0, 0, 0, 0, 1, 0, 1},
    {0, 0, 0, 0, 0, 0, 1, 0}
};
```

2. Display the adjacency matrix. Put a bracket to the points/destinations that are considered as charging stations, e.g. [c], [d]

```
for (int i = 0; i < NUM_POINTS; i++) {
    for (int j = 0; j < NUM_POINTS; j++) {
        // Check if the current point is a charging station
        // c or d
        if (i == 2 || i == 3) {
            printf("[%d]", road_networks[i][j]);
        } else {
            printf("%d", road_networks[i][j]);
        }
    }
}
```

```

    }
    printf("\n");
}

```

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.

```

int nearest_charging_station(int point) {
    int min_distance = INT_MAX;
    int nearest_cs = -1;

    // Loop through all charging stations (c and d)
    for (int i = 2; i <= 3; i++) {
        // Check if there is a direct path between the current point and the charging station
        if (road_networks[point][i]) {
            // Calculate the distance between the current point and the charging station
            int distance = 1; // Assume that all direct paths have a distance of 1
            if (distance < min_distance) {
                min_distance = distance;
                nearest_cs = i;
            }
        }
    }
}

```

4. Bonus: Use a macro to define the size of the 2d array

```

#define NUM_POINTS 8

```

### ***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`

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