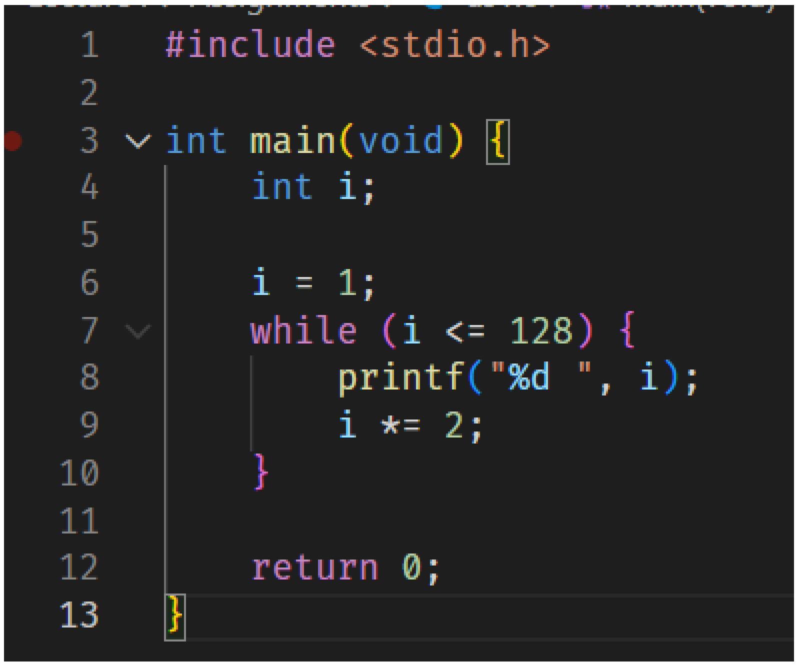
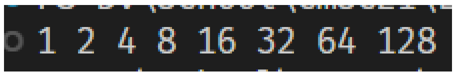
Ezra Magbanua
202206040

1. What is the output of the following program?



```c
#include <stdio.h>

int main(void) {
    int i;

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

    return 0;
}
```

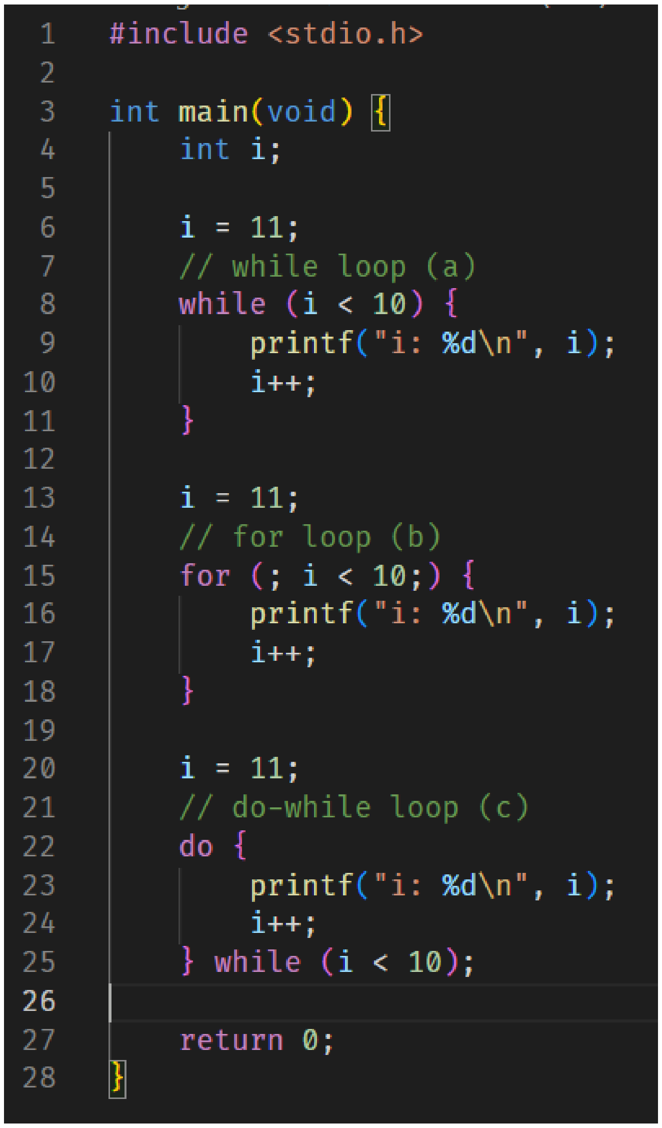


```
1 2 4 8 16 32 64 128
```

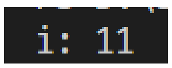
The code prints 2 to the n, where n starts from 0 and ends at 7 (1 2 4 8 16 32 64 128).

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);



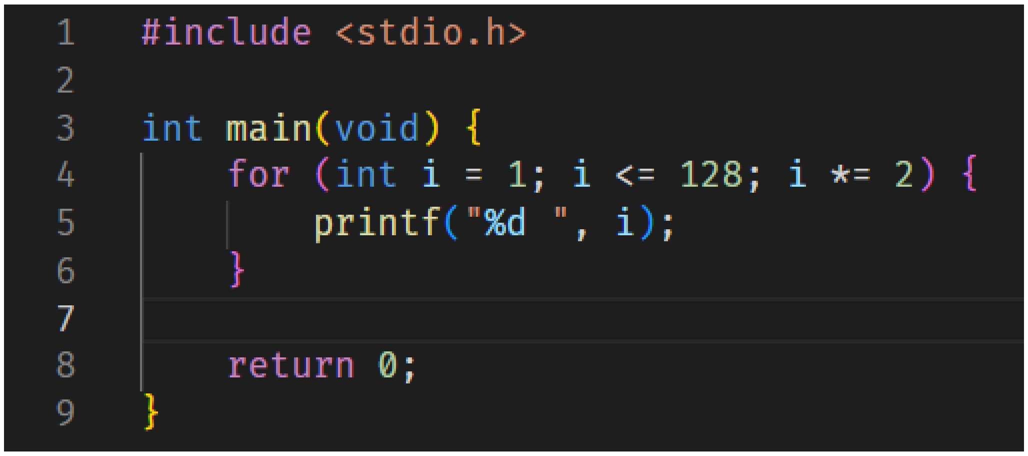
```c
#include <stdio.h>

int main(void) {
    int i;

    i = 11;
    // while loop (a)
    while (i < 10) {
        printf("i: %d\n", i);
        i++;
    }

    i = 11;
    // for loop (b)
    for (; i < 10;) {
        printf("i: %d\n", i);
        i++;
    }

    i = 11;
    // do-while loop (c)
    do {
        printf("i: %d\n", i);
        i++;
    } while (i < 10);

    return 0;
}
```



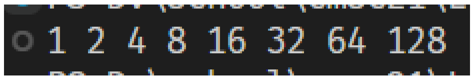
```
i: 11
```

The last loop (c / do while loop) is not equivalent to the other two since it executes the loop body first without checking the statement first.

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.

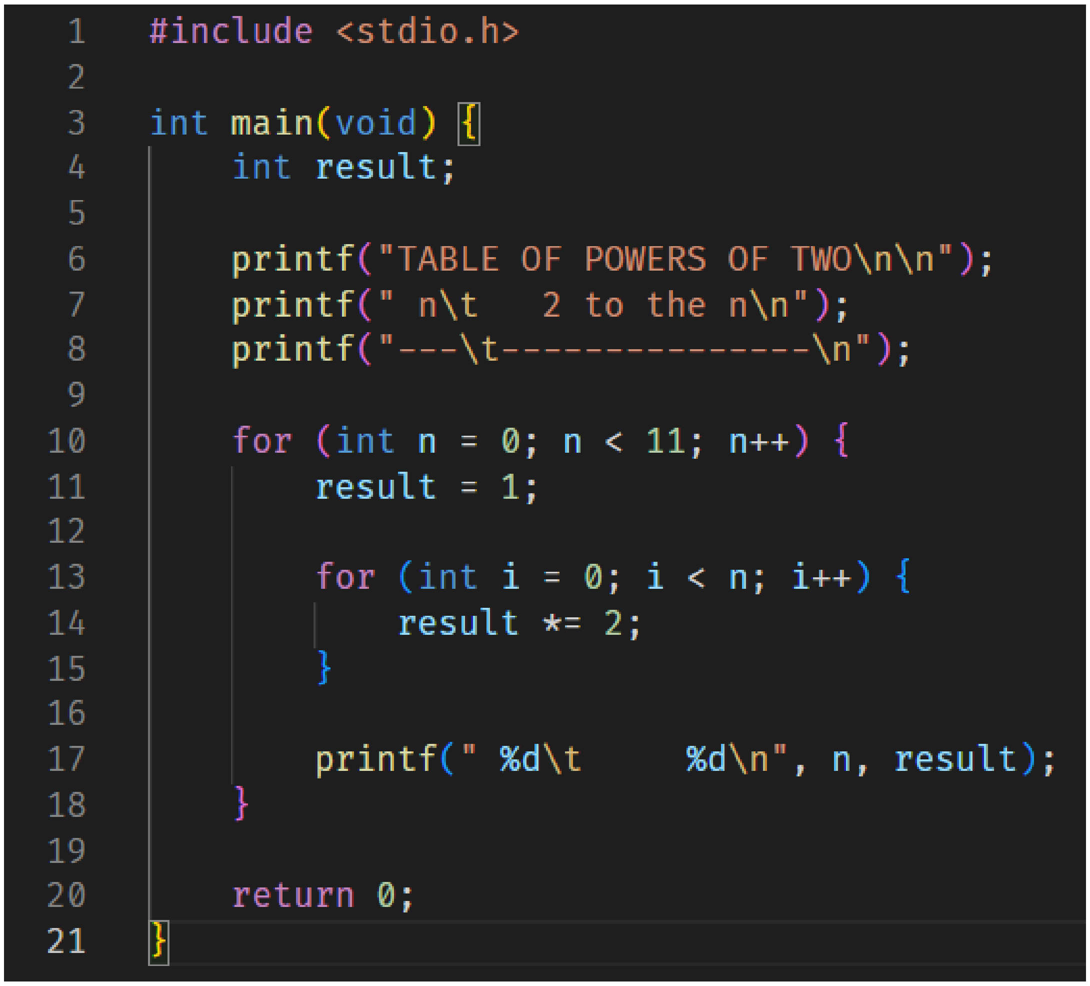


```c
#include <stdio.h>

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

    return 0;
}
```

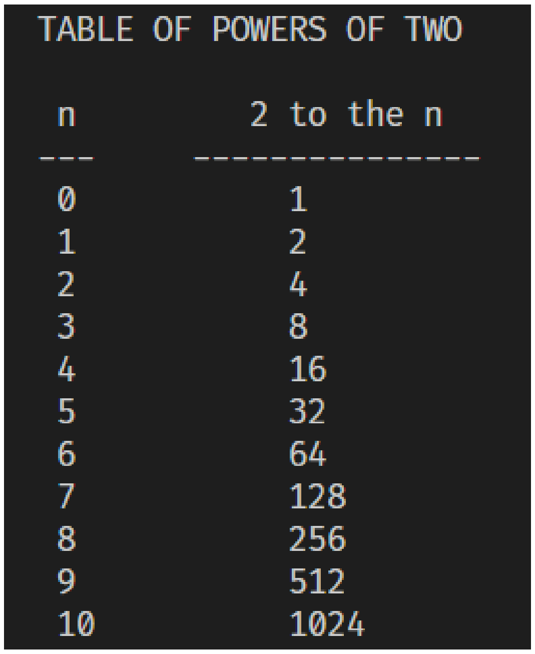


```
1 2 4 8 16 32 64 128
```

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

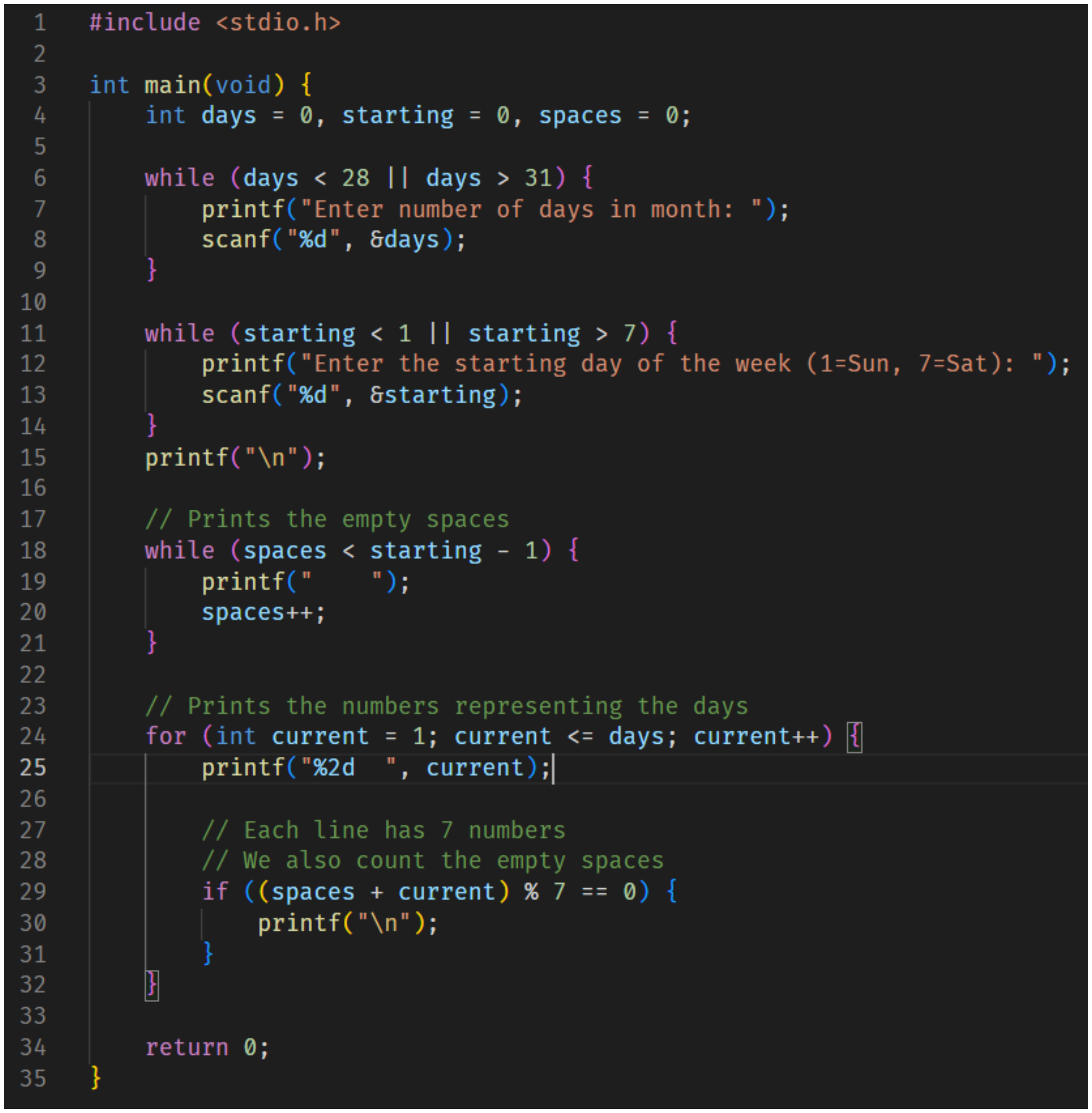


```c
#include <stdio.h>

int main(void) {
    int result;

    printf("TABLE OF POWERS OF TWO\n\n");
    printf(" n\t   2 to the n\n");
    printf("---\t---------------\n");

    for (int n = 0; n < 11; n++) {
        result = 1;

        for (int i = 0; i < n; i++) {
            result *= 2;
        }

        printf(" %d\t     %d\n", n, result);
    }

    return 0;
}
```

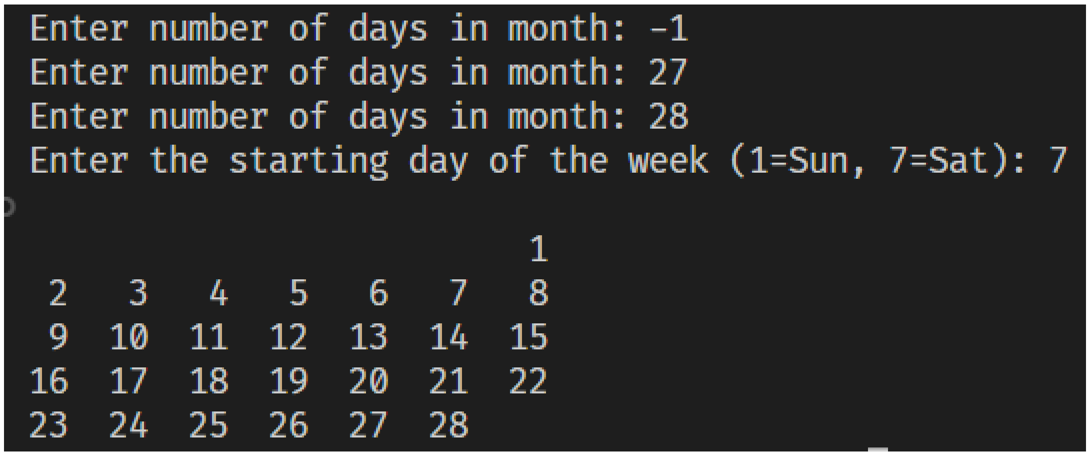


```
TABLE OF POWERS OF TWO

 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
```

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



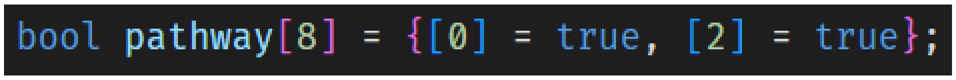
```c
#include <stdio.h>

int main(void) {
    int days = 0, starting = 0, spaces = 0;

    while (days < 28 || days > 31) {
        printf("Enter number of days in month: ");
        scanf("%d", &days);
    }

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

    // Prints the empty spaces
    while (spaces < starting - 1) {
        printf("    ");
        spaces++;
    }

    // Prints the numbers representing the days
    for (int current = 1; current <= days; current++) {
        printf("%2d  ", current);

        // Each line has 7 numbers
        // We also count the empty spaces
        if ((spaces + current) % 7 == 0) {
            printf("\n");
        }
    }

    return 0;
}
```



```
Enter number of days in month: -1
Enter number of days in month: 27
Enter number of days in month: 28
Enter the starting day of the week (1=Sun, 7=Sat): 7

                         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
```

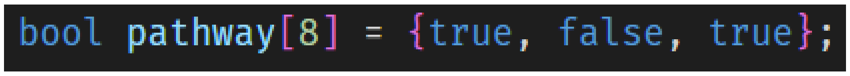
6.

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};
```

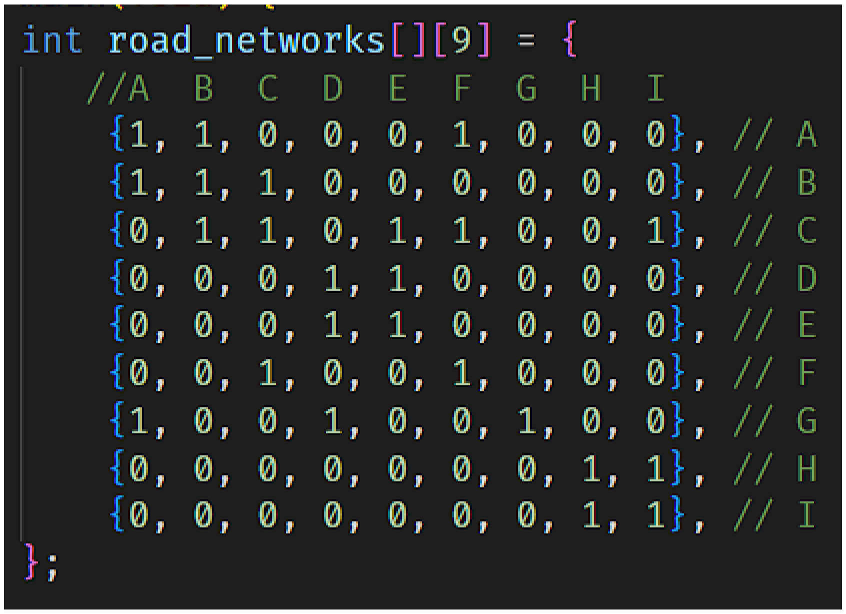
b) Revise line 16 such that the initializer will be short as possible (without using a designated initializer)



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

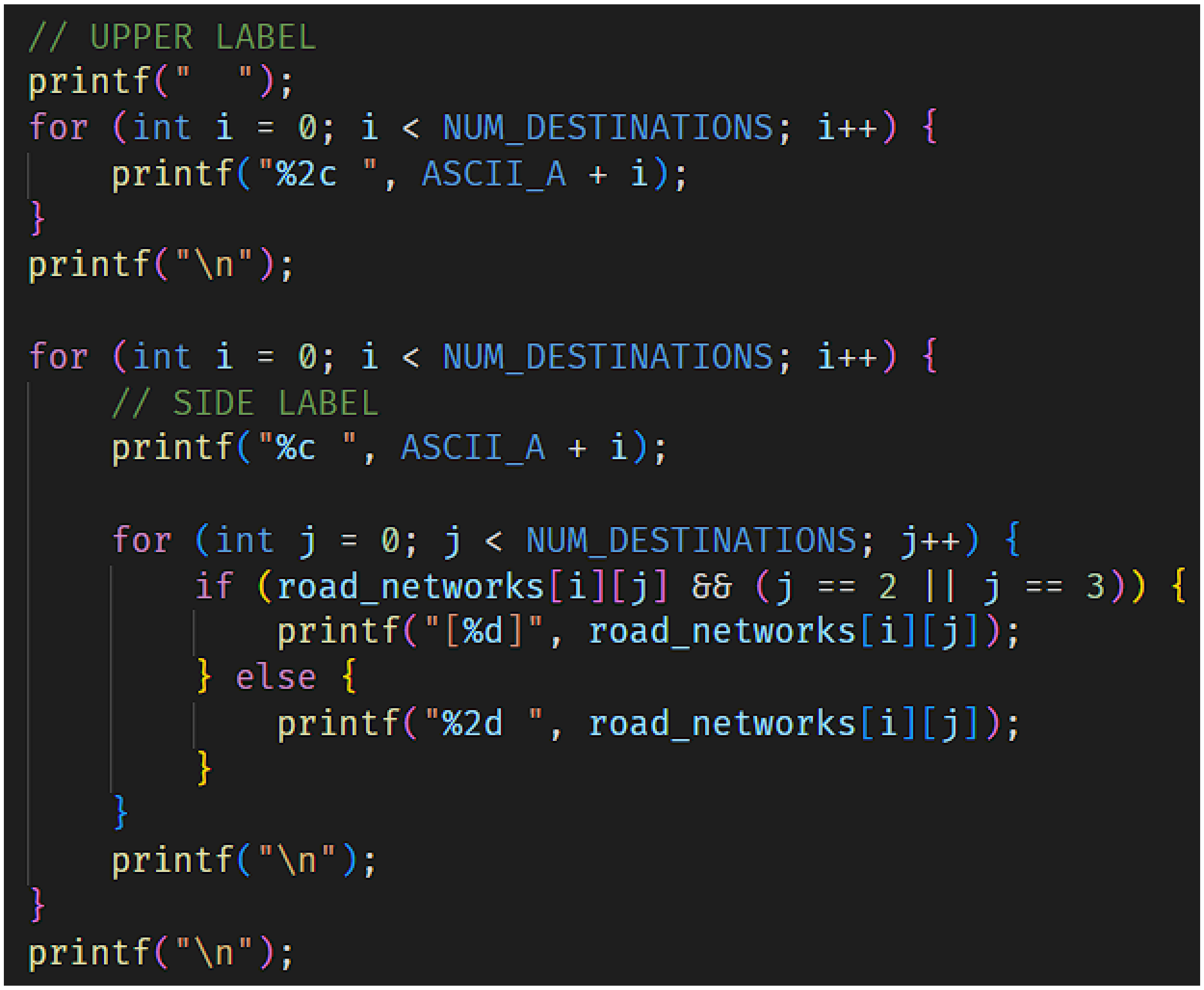
7.

a) Declare and initialize a road_networks multidimensional array that represents the adjacency matrix

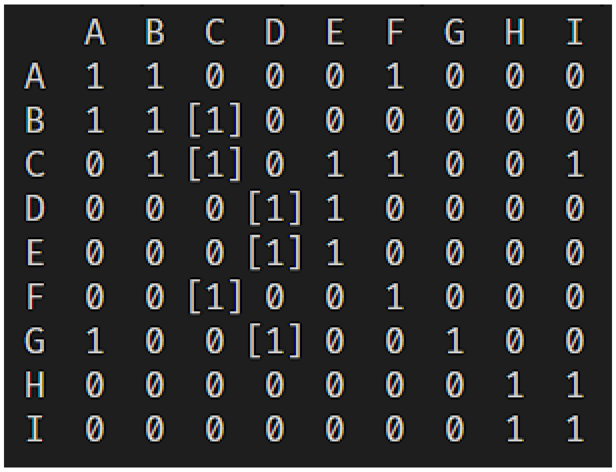


```
int road_networks[][9] = {
   //A  B  C  D  E  F  G  H  I
    {1, 1, 0, 0, 0, 1, 0, 0, 0}, // A
    {1, 1, 1, 0, 0, 0, 0, 0, 0}, // B
    {0, 1, 1, 0, 1, 1, 0, 0, 1}, // C
    {0, 0, 0, 1, 1, 0, 0, 0, 0}, // D
    {0, 0, 0, 1, 1, 0, 0, 0, 0}, // E
    {0, 0, 1, 0, 0, 1, 0, 0, 0}, // F
    {1, 0, 0, 1, 0, 0, 1, 0, 0}, // G
    {0, 0, 0, 0, 0, 0, 0, 1, 1}, // H
    {0, 0, 0, 0, 0, 0, 0, 1, 1}, // I
};
```

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

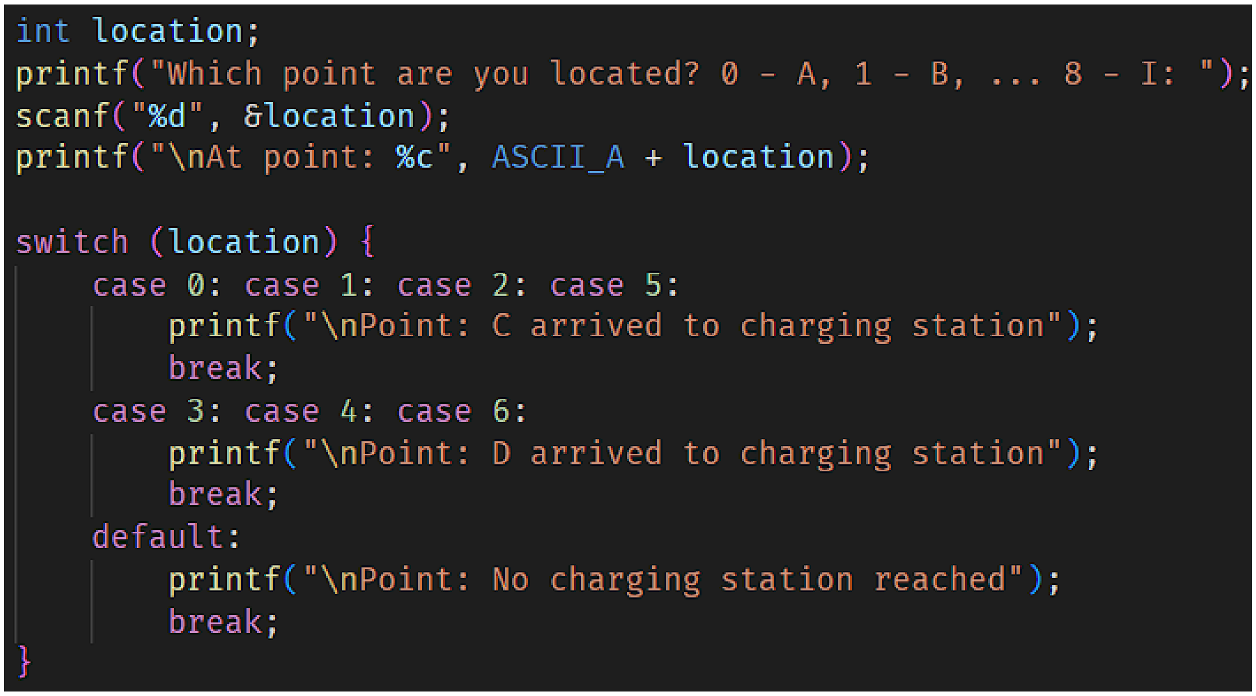


```c
// UPPER LABEL
printf("   ");
for (int i = 0; i < NUM_DESTINATIONS; i++) {
    printf("%2c ", ASCII_A + i);
}
printf("\n");

for (int i = 0; i < NUM_DESTINATIONS; i++) {
    // SIDE LABEL
    printf("%c ", ASCII_A + i);

    for (int j = 0; j < NUM_DESTINATIONS; j++) {
        if (road_networks[i][j] && (j == 2 || j == 3)) {
            printf("[%d]", road_networks[i][j]);
        } else {
            printf("%2d ", road_networks[i][j]);
        }
    }
    printf("\n");
}
printf("\n");
```

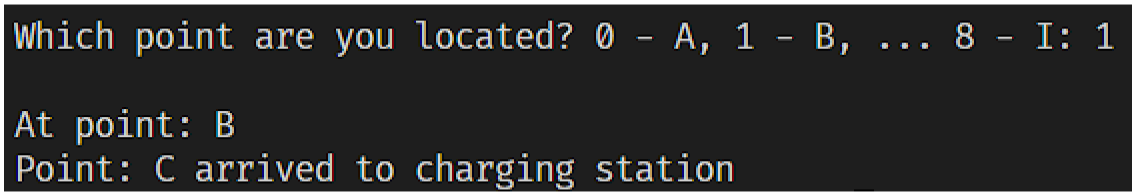


```
    A  B  C  D  E  F  G  H  I
A   1  1  0  0  0  1  0  0  0
B   1  1 [1] 0  0  0  0  0  0
C   0  1 [1] 0  1  1  0  0  1
D   0  0  0 [1] 1  0  0  0  0
E   0  0  0 [1] 1  0  0  0  0
F   0  0 [1] 0  0  1  0  0  0
G   1  0  0 [1] 0  0  1  0  0
H   0  0  0  0  0  0  0  1  1
I   0  0  0  0  0  0  0  1  1
```

c.) 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.

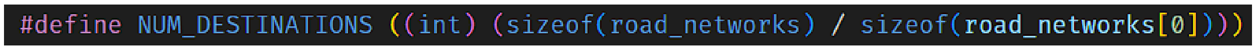


```c
int location;
printf("Which point are you located? 0 - A, 1 - B, ... 8 - I: ");
scanf("%d", &location);
printf("\nAt point: %c", ASCII_A + location);

switch (location) {
    case 0: case 1: case 2: case 5:
        printf("\nPoint: C arrived to charging station");
        break;
    case 3: case 4: case 6:
        printf("\nPoint: D arrived to charging station");
        break;
    default:
        printf("\nPoint: No charging station reached");
        break;
}
```



```
Which point are you located? 0 - A, 1 - B, ... 8 - I: 1

At point: B
Point: C arrived to charging station
```

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



```c
#define NUM_DESTINATIONS ((int) (sizeof(road_networks) / sizeof(road_networks[0])))
```