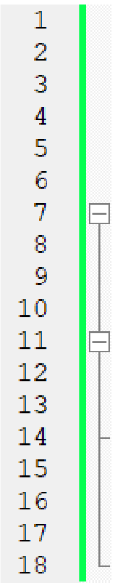
Neil Bryan F. Flauta   202151001
CMSC 21 – 1   April 3, 2022

Loop/Repetition Statements
Lecture 4 Assignments

1. What is the output of the following program?

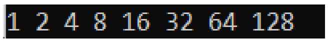
```
#include <stdio.h>

int main(void)
{
   int i;

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

   return 0;
}
```

Screenshot of the code:



```
//Lecture 4 Assignment
//No. 1

#include <stdio.h>

int main(void)
{
    int i;

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

    return 0;
}
```

Output:

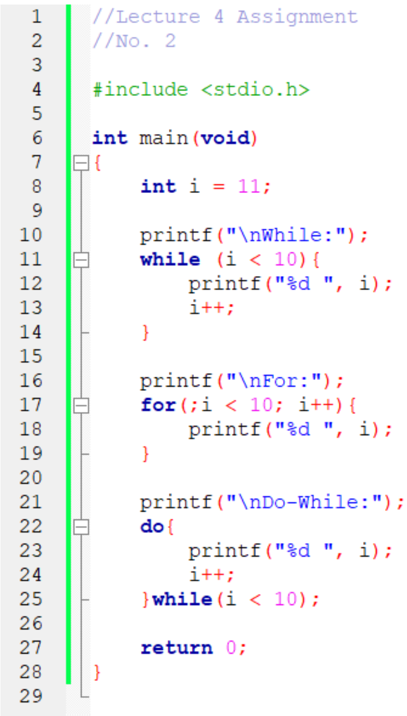


```
1 2 4 8 16 32 64 128
```

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

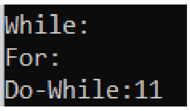
```
a) while (i < 10) {...}
b) for (; i < 10;) {...}
c) do {...} while (i < 10);
```

Screenshot of the code:



```
//Lecture 4 Assignment
//No. 2

#include <stdio.h>

int main(void)
{
    int i = 11;

    printf("\nWhile:");
    while (i < 10){
        printf("%d ", i);
        i++;
    }

    printf("\nFor:");
    for(;i < 10; i++){
        printf("%d ", i);
    }

    printf("\nDo-While:");
    do{
        printf("%d ", i);
        i++;
    }while(i < 10);

    return 0;
}
```

Output:



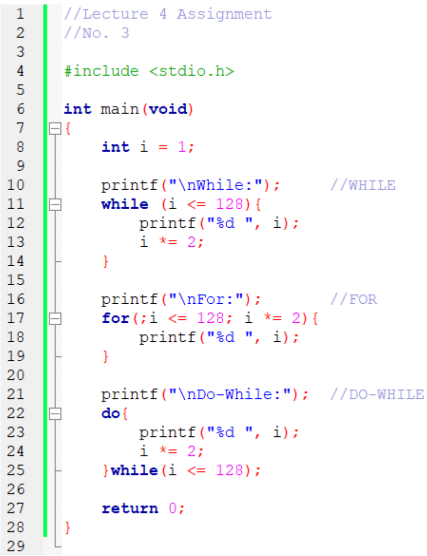
```
While:
For:
Do-While:11
```

Explanation:

Among the three statements, the ***Do-While*** statement is not equivalent to the other two. It is because in the ***Do-While,*** the execution of the statement(s) in the loop body comes first before the evaluation of the condition, unlike in the ***While*** and ***For***. In the screenshot of the code, the loop control variable ***i*** is set to '11' to make the condition in each statement wrong. This will lead to termination of each loop (since 11 > 10) but **Do-While** is an exception because the statements(s) are guaranteed to execute in the first iteration.

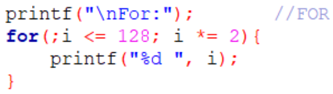
3. Convert item 1 into an equivalent for statement. You can validate your answer by checking if the produced outputs by both the while and for statements are similar.

Screenshot of the code:



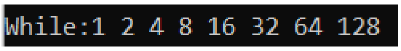
```
//Lecture 4 Assignment
//No. 3

#include <stdio.h>

int main(void)
{
    int i = 1;

    printf("\nWhile:");     //WHILE
    while (i <= 128){
        printf("%d ", i);
        i *= 2;
    }

    printf("\nFor:");       //FOR
    for(;i <= 128; i *= 2){
        printf("%d ", i);
    }

    printf("\nDo-While:");  //DO-WHILE
    do{
        printf("%d ", i);
        i *= 2;
    }while(i <= 128);

    return 0;
}

```

Equivalent FOR statement:

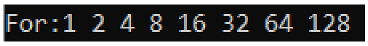


```
printf("\nFor:");       //FOR
for(;i <= 128; i *= 2){
    printf("%d ", i);
}
```

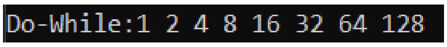
Outputs of each statement:
(As each loop statement are ran one by one, the other two loop statements are converted to comments)



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



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



```
Do-While:1 2 4 8 16 32 64 128
```

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

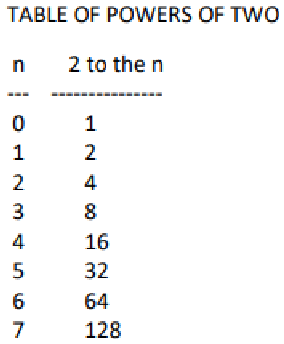
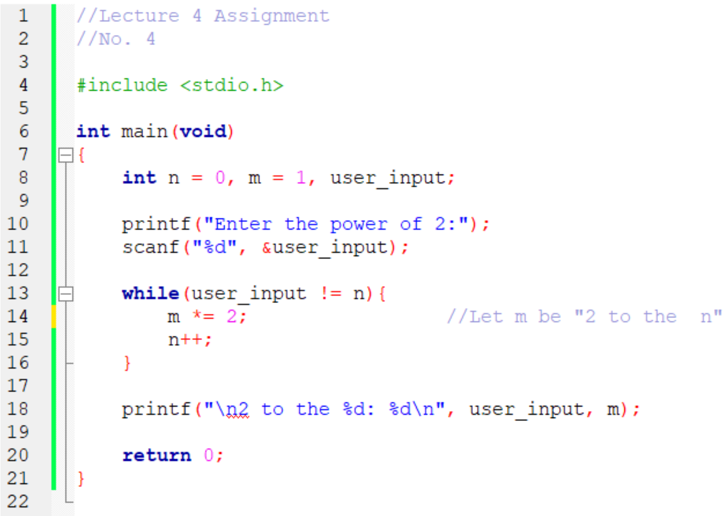


TABLE OF POWERS OF TWO

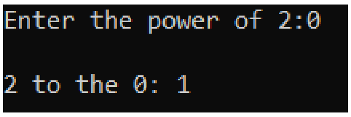
n	2 to the n
---	---------------
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128

Screenshot of the code:

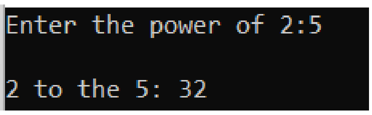


```
//Lecture 4 Assignment
//No. 4

#include <stdio.h>

int main(void)
{
    int n = 0, m = 1, user_input;

    printf("Enter the power of 2:");
    scanf("%d", &user_input);

    while(user_input != n){
        m *= 2;                 //Let m be "2 to the  n"
        n++;
    }

    printf("\n2 to the %d: %d\n", user_input, m);

    return 0;
}

```

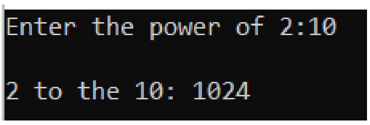
Example Outputs:



```
Enter the power of 2:0

2 to the 0: 1
```

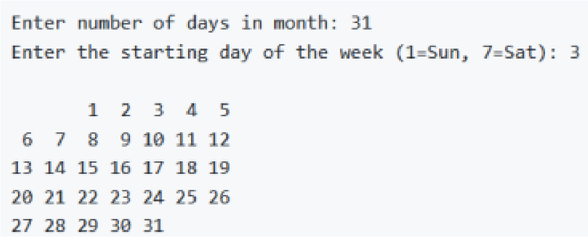


```
Enter the power of 2:5

2 to the 5: 32
```



```
Enter the power of 2:10

2 to the 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
```

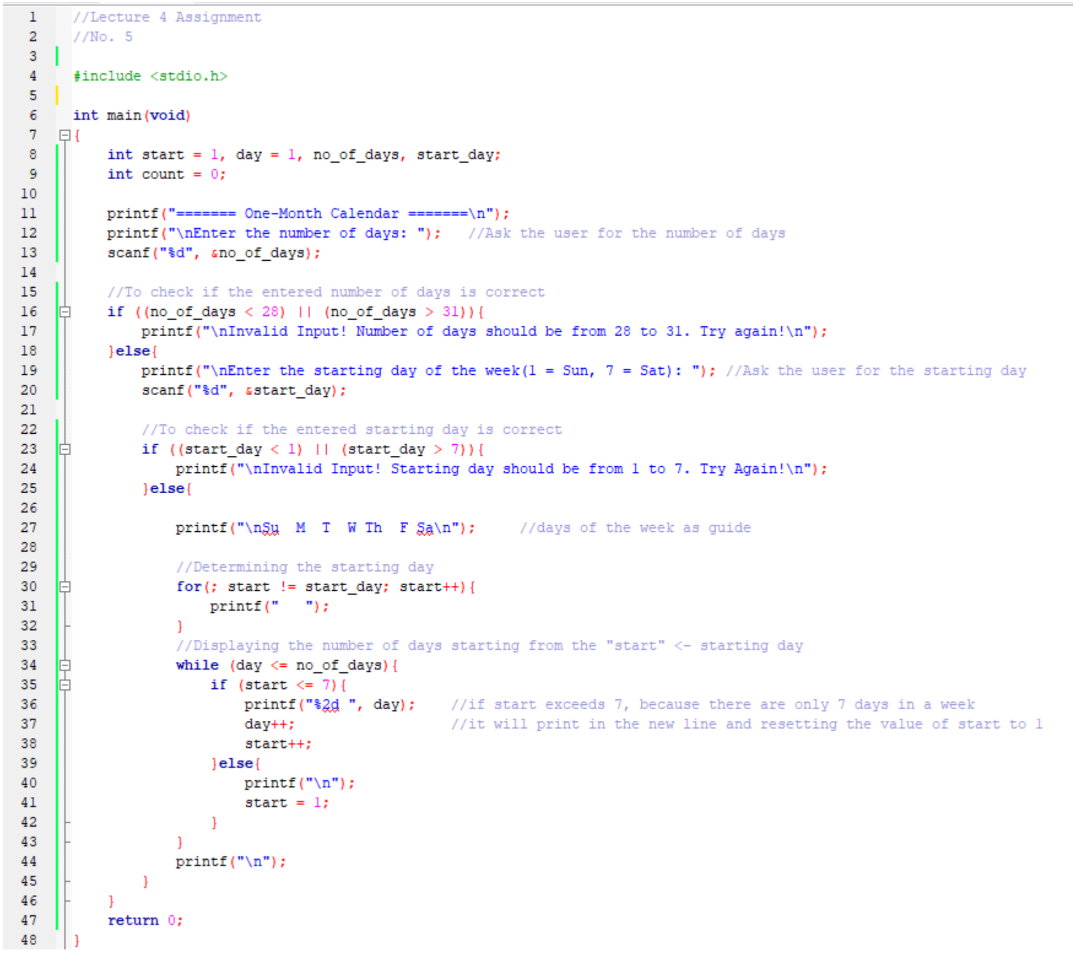
There should be a user prompt to set:

- 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.

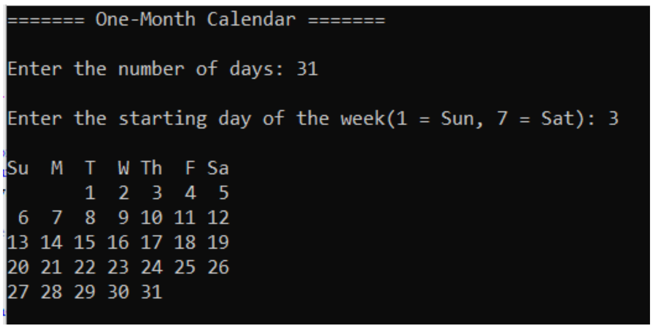
Screenshot of the code:



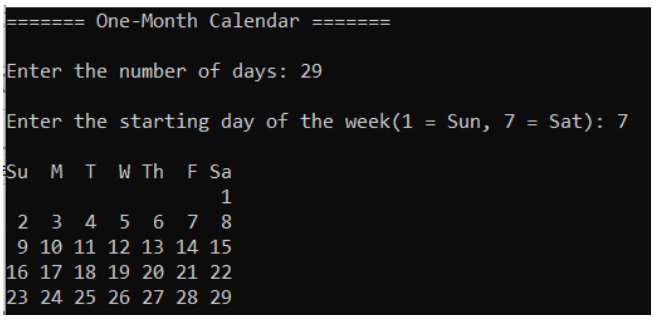
```c
//Lecture 4 Assignment
//No. 5

#include <stdio.h>

int main(void)
{
    int start = 1, day = 1, no_of_days, start_day;
    int count = 0;

    printf("======= One-Month Calendar =======\n");
    printf("\nEnter the number of days: ");   //Ask the user for the number of days
    scanf("%d", &no_of_days);

    //To check if the entered number of days is correct
    if ((no_of_days < 28) || (no_of_days > 31)){
        printf("\nInvalid Input! Number of days should be from 28 to 31. Try again!\n");
    }else{
        printf("\nEnter the starting day of the week(1 = Sun, 7 = Sat): "); //Ask the user for the starting day
        scanf("%d", &start_day);

        //To check if the entered starting day is correct
        if ((start_day < 1) || (start_day > 7)){
            printf("\nInvalid Input! Starting day should be from 1 to 7. Try Again!\n");
        }else{

            printf("\nSu  M  T  W Th  F Sa\n");     //days of the week as guide

            //Determining the starting day
            for(; start != start_day; start++){
                printf("   ");
            }
            //Displaying the number of days starting from the "start" <- starting day
            while (day <= no_of_days){
                if (start <= 7){
                    printf("%2d ", day);    //if start exceeds 7, because there are only 7 days in a week
                    day++;                  //it will print in the new line and resetting the value of start to 1
                    start++;
                }else{
                    printf("\n");
                    start = 1;
                }
            }
            printf("\n");
        }
    }
    return 0;
}
```

Example Outputs:

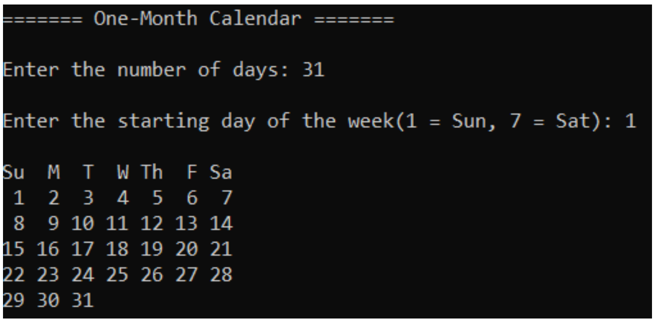
Correct Inputs:



```
======= One-Month Calendar =======

Enter the number of days: 31

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

Su  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
```

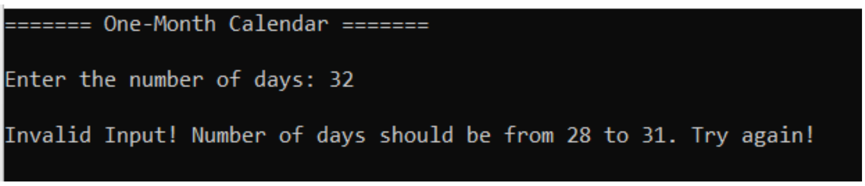


```
======= One-Month Calendar =======

Enter the number of days: 29

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

Su  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
```



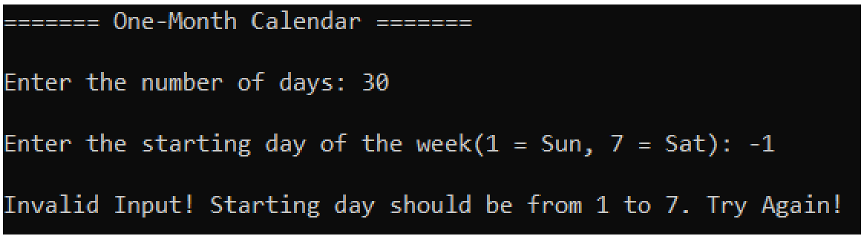
```
======= One-Month Calendar =======

Enter the number of days: 31

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

Su  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
```

Incorrect Input of number of days (28 – 31 days only)



```
======= One-Month Calendar =======

Enter the number of days: 32

Invalid Input! Number of days should be from 28 to 31. Try again!
```

Incorrect input of starting day (1 – 7 only)



```
======= One-Month Calendar =======

Enter the number of days: 30

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

Invalid Input! Starting day should be from 1 to 7. Try Again!
```

GitHub Link: https://github.com/nbbryy/CMSC-21.git