Loop/Repetition Statements
Lecture 4 Assignments

1.

```c
/***  CELIS, KRISTINA | ASSIGNMENT 4 | as1  ***/

#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
PS C:\Users\Kristina\Desktop\ACADS\CMSC 21\source codes\lecture 4>
```

Output:

1 2 4 8 16 32 64 128

2.

```c
/***  CELIS, KRISTINA | ASSIGNMENT 4 | as2  ***/

#include <stdio.h>

int main(void) {
    int i, num;

    printf("Enter an integer (i): ");   //input will be the test variable for the loops
    scanf("%d", &num);

    i = num;
    printf("::: WHILE :::\n");
    while (i < 10) {
        printf("%d ", i);
        i++;
    }

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

    i = num;
    printf("\n::: DO-WHILE :::\n");
    do {
        printf("%d ", i);
        i++;
    } while (i < 10);

    return 0;
}
```

Explanation:

The three loops will display the same output if the condition (i < 10) is TRUE. So, in the first output with an input of 1, we have all the 3 loops printing the same values. In the second output with an input of 11, the while and for loops didn't produce anything because it didn't meet the condition i < 10. However, the do-while loop produced a value of 11 because its main purpose is to execute the body at least once even if the condition was not satisfied. Hence, the **do-while** loop is the one that is not equivalent to the other two.

```
Enter an integer (i): 1
::: WHILE :::
1 2 3 4 5 6 7 8 9
::: FOR :::
1 2 3 4 5 6 7 8 9
::: DO-WHILE :::
1 2 3 4 5 6 7 8 9
PS C:\Users\Kristina\Desktop\ACADS\CMSC 21\source codes\lecture 4>
```

```
Enter an integer (i): 11
::: WHILE :::

::: FOR :::

::: DO-WHILE :::
11
PS C:\Users\Kristina\Desktop\ACADS\CMSC 21\source codes\lecture 4>
```

GitHub Link: https://github.com/tinapayy/CMSC21.git

3.

```c
/***  CELIS, KRISTINA | ASSIGNMENT 4 | as3  ***/

#include <stdio.h>

int main(void) {
    int i;

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

    return 0;
}
```

```
1 2 4 8 16 32 64 128
PS C:\Users\Kristina\Desktop\ACADS\CMSC 21\source codes\lecture 4>
```

4.

```c
/***  CELIS, KRISTINA | ASSIGNMENT 4 | as4  ***/

#include <stdio.h>

int main(void) {
    /* VARIABLES */
    int input, n, two_n;

    /* INPUT */
    printf("Enter (n): ");
    scanf("%d", &input);

    /* OUTPUT */
    printf("  n   2 to the n\n ---  -----------\n");
    two_n = 1;
    for (n = 0; n <= input; n++) {
        printf("%3d      %d\n", n, two_n);      // %3d --> right-justified; spaces before %d to display clean output
        two_n *= 2;
    }

    return 0;
}
```

```
Enter (n): 10
  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
PS C:\Users\Kristina\Desktop\ACADS\CMSC 21\source codes\lecture 4>
```

GitHub Link: https://github.com/tinapayy/CMSC21.git

5.

```c
/***  CELIS, KRISTINA | ASSIGNMENT 4 | as5  ***/

#include <stdio.h>

int main(void) {
    /* VARIABLES */
    int days, n_day, slot;

    /* INPUT & VALIDATION */
    while (1) {
        printf("Enter number of days in the month (28-31 only): ");
        scanf("%d", &days);

        if (days < 28 || days > 31) {       // invalid
            printf("Invalid Input! Must be 28 to 31 only. Try again.\n");
        }
        else {      // valid
            break;
        }
    }

    while (1) {
        printf("Enter the starting day of the week (1 = Sun, 7 = Sat): ");
        scanf("%d", &n_day);

        if (n_day < 1 || n_day > 7) {
            printf("Invalid Input! Must be 1 to 7 only. Try again.");
        }
        else {
            break;
        }
    }
    /* OUTPUT */
    printf("\n~~~C A L E N D A R~~~\n");
    printf("\nSu Mo Tu We Th Fr Sa\n");

    for (slot = 1; slot < n_day; slot++) {      // prints spaces depending on the weekday chosed
        printf("   ");
    }
    n_day = (8 - n_day) % 7;                    // will be used to indicate end rows
    for (slot = 1; slot <= days; slot++) {      // assigns the dates in the corresponding weekdays until no slot left
        printf("%2d ", slot);

        if (n_day == slot % 7) {                // when saturday or row has ended, we print a new line
            printf("\n");
        }
    }
    return 0;
}
```

```
Enter number of days in the month (28-31 only): 28
Enter the starting day of the week (1 = Sun, 7 = Sat): 4

~~~C A L E N D A R~~~

Su Mo Tu We Th Fr 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\Kristina\Desktop\ACADS\CMSC 21\source codes\lecture 4>
```

```
Enter number of days in the month (28-31 only): 31
Enter the starting day of the week (1 = Sun, 7 = Sat): 7

~~~C A L E N D A R~~~

Su Mo Tu We Th Fr 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\Kristina\Desktop\ACADS\CMSC 21\source codes\lecture 4>
```

```
Enter number of days in the month (28-31 only): 21
Invalid Input! Must be 28 to 31 only. Try again.
Enter number of days in the month (28-31 only): 7
Invalid Input! Must be 28 to 31 only. Try again.
Enter number of days in the month (28-31 only): 0
Invalid Input! Must be 28 to 31 only. Try again.
Enter number of days in the month (28-31 only): 32
Invalid Input! Must be 28 to 31 only. Try again.
Enter number of days in the month (28-31 only):
```

GitHub Link: https://github.com/tinapayy/CMSC21.git