

Christian Justin J. Salinas – 2021-50521
Instructor Ara Abigail Ambita
CMSC 21 – 2
Lecture 4 Assignment

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;
}
```

Complete source code:

```
1  /*
2      Lecture 4 Assignment #1
3      Checking the output of the following program
4      By: Christian Justin J. Salinas
5  */
6
7  #include <stdio.h>
8
9  int main(void)
10 {
11     int i;
12
13     i = 1;
14     while (i <= 128){
15         printf("%d ", i);
16         i *= 2;
17     }
18
19     return 0;
20 }
21
```

Output of the program above: 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);

Complete source code:

```
7  #include <stdio.h>
8
9  int main() {
10     int i = 11; // As an example
11
12     /* While Loop */
13     while (i < 10){
14         printf("%d ", i);
15         i++;
16     }
17     return 0;
18 }
19
```

```
7  #include <stdio.h>
8
9  int main() {
10     int i = 11; // As an example
11
12     /* For Loop */
13     for (; i < 10; i++){
14         printf("%d ", i);
15     }
16     return 0;
17 }
18
19
```

```
7  #include <stdio.h>
8
9  int main() {
10     int i = 11; // As an example
11
12     /* Do-While Loop */
13     do{
14         printf("%d ", i);
15         i++;
16     }while(i < 10);
17     return 0;
18 }
19
20
```

Outputs:

- While Loop:
- For Loop:
- Do-While Loop: 11

Explanation: The **Do-While** statement in Letter C is not equivalent to the other two (A's **while** and B's **for**). Because the execution of the statements in the **Do-While** comes first before the evaluation of the condition ($i < 10$). In contrast, **While** and **For** loops both start with the evaluation of the condition. To test this, i was declared as an integer type and with 11 assigned to it to make the conditions ($i < 10$) false. The **While** and **For** will not output anything because of the condition set, but **Do-While** will output the first iteration before the evaluation of the condition.

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.

Complete source code:

```

7  #include <stdio.h>
8
9  int main() {
10     int i = 1;
11
12     /* While Loop */
13     while (i <= 128) {
14         printf("%d ", i);
15         i *= 2;
16     }
17
18     /* For Loop */
19     for (; i <= 128; i *= 2) {
20         printf("%d ", i);
21     }
22
23     /* Do-While Loop */
24     do {
25         printf("%d ", i);
26         i *= 2;
27     } while (i <= 128);
28
29     return 0;
30 }
31

```

FOR statement that is equivalent to item 1:

```

7  #include <stdio.h>
8
9  int main() {
10     int i = 1;
11
12     /* For Loop */
13     for (; i <= 128; i *= 2) {
14         printf("%d ", i);
15     }
16
17     return 0;
18 }
19

```

Outputs of each loop statement:

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

Note: Each statement has been evaluated separately with the use of comments for convenience.

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

Complete source code:

```

7  #include <stdio.h>
8
9  int main() {
10     /* Declare variables by int type */
11     int n = 0, m = 1, user_input;
12
13     /* Ask for user to input a power for 2 and store it */
14     printf("Enter the power of 2: ");
15     scanf("%d", &user_input);
16
17     /* Evaluate power for user input */
18     while(user_input != n){
19         m *= 2; // Formula: Let m be "2 to the n"
20         n++; // Increment by 1
21     }
22
23     /* Display output by input and output */
24     printf("\n2 to the %d is %d\n", user_input, m);
25
26     /* Indicate that the program ends here */
27     return 0;
28 }
29

```

Sample outputs:

Test 1

```

Enter the power of 2: 7
2 to the 7 is 128

```

Test 2

```

Enter the power of 2: 3
2 to the 3 is 8

```

Test 3

```

Enter the power of 2: 4
2 to the 4 is 16

```

Test 4

```

Enter the power of 2: 10
2 to the 10 is 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
```

```
7  #include <stdio.h> // For getting inputs and outputs properly
8
9  int main() {
10     /* Declare variables by int type */
11     int i, j, days_in_month, day_of_week;
12
13     /* Ask the user for number of days and store it */
14     printf ("Enter number of days in month: ");
15     scanf ("%d", &days_in_month);
16
17     /* Check if days in month is valid, notify the user if not */
18     if ((days_in_month < 28) || (days_in_month > 31)) {
19         printf("Invalid input: Please enter valid number of days in a month from 28 to 31.\n");
20     } else {
21         /* Ask the user for day of week and store it */
22         printf ("Enter the starting day of the week (1=Sun, 7=Sat): ");
23         scanf ("%d", &day_of_week);
24
25         /* Check if day of week is valid, notify the user if not */
26         if ((day_of_week < 1) || (day_of_week > 7)) {
27             printf("Invalid input: Please enter valid day of a week from 1 to 7.\n");
28         } else {
29             // Let day of week be 0-6 rather than 1-7
30             day_of_week -= 1;
31
32             // Print days of the week
33             printf("\n  S  M  T  W  Th  F  S\n");
34
35             // Print the first line of the calendar (readability)
36             for(i = 0; i < day_of_week; i++)
37                 printf("    ");
38
39             // For each day in the month...
40             for(i = 1; i <= days_in_month; i++) {
41                 /* Print the current day of the week.
42                  * At least 3 spaces ("%3d") are inserted with the value */
43                 printf("%3d", i);
44
45                 /* Increment the day_of_week.
46                  * And wrap day_of_week around to 0 with the modulo operator '% 7' when it reaches 7. */
47                 day_of_week = (day_of_week + 1) % 7;
48
49                 /* If the new day_of_week is 0, output a newline to start at the
50                  * beginning of the next line */
51                 if(day_of_week == 0)
52                     printf("\n");
53             }
54         }
55     }
56     /* Indicate that the program ends here */
57     return 0;
58 }
59
```

Sample outputs:

With valid inputs:

```
Enter number of days in month: 31
Enter the starting day of the week (1=Sun, 7=Sat): 3

  S  M  T  W Th  F  S
        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
```

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

  S  M  T  W Th  F  S
  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
```

```
Enter number of days in month: 30
Enter the starting day of the week (1=Sun, 7=Sat): 7

  S  M  T  W Th  F  S
        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
```

With invalid inputs:

```
Enter number of days in month: 25
Invalid input: Please enter valid number of days in a month from 28 to 31.
```

```
Enter number of days in month: 31
Enter the starting day of the week (1=Sun, 7=Sat): 8
Invalid input: Please enter valid day of a week from 1 to 7.
```

```
Enter number of days in month: 31
Enter the starting day of the week (1=Sun, 7=Sat): -1
Invalid input: Please enter valid day of a week from 1 to 7.
```