

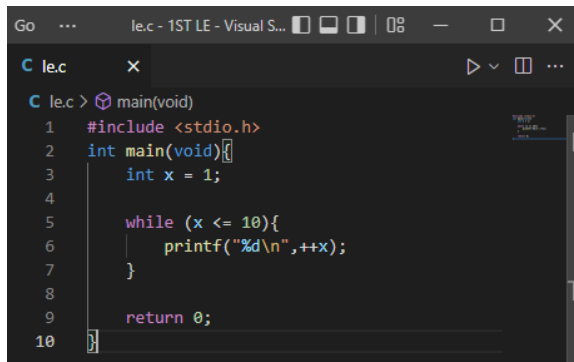
## CMSC 21 First Long Examination

Part I. True or False. Justify your answer.

1. True, because a long double's storage size is 10 bytes while the double is only 8 bytes.
2. True, zero is interpreted false while one is interpreted as true so all values that are not zero is considered as true.
3. False, == is used for comparison while = is used for assignment of two quantities.
4. True, because keywords are reserved words that cannot be used as variables.
5. True, the ampersand (&) is used to retrieve the address of value and store it to a variable.
6. False, sign qualifiers are not applicable for bool type of data.
7. False, because ((a == b) || (b > a)) && (d < a) is true && true and that will generate a true.
8. False, the break statement is used to terminate a switch statement and is not needed in the default because if it is the last statement.
9. False, both statements should be true because it uses an && operator.
10. False, because the first one has a condition that needs to be satisfied.

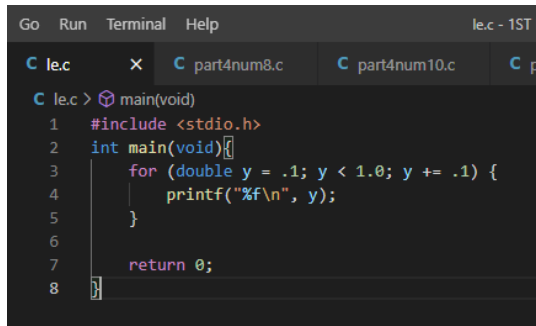
Part II. Find the errors in the following program. Indicate the possible correction.

1. The code should have braces in the while loop and the while loop body should have a print statement.



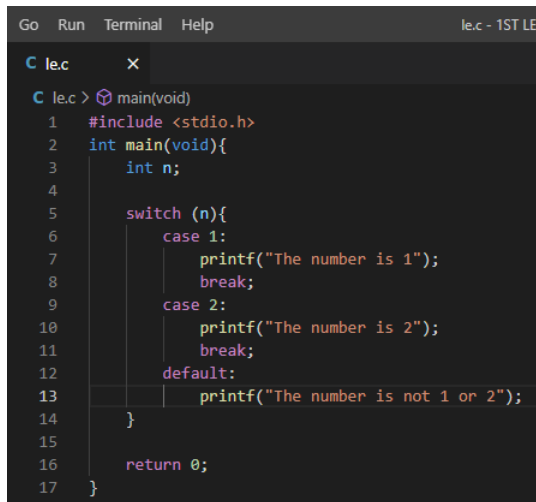
```
le.c > main(void)
1  #include <stdio.h>
2  int main(void){
3      int x = 1;
4
5      while (x <= 10){
6          printf("%d\\n", ++x);
7      }
8
9      return 0;
10 }
```

2. The second statement should be  $y < 1.0$  so that it will print the values from 0.100000 to 1.000000.



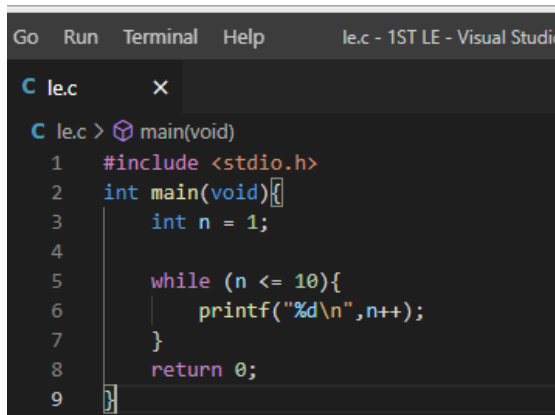
```
le.c > main(void)
1  #include <stdio.h>
2  int main(void){
3      for (double y = .1; y < 1.0; y += .1) {
4          printf("%f\\n", y);
5      }
6
7      return 0;
8  }
```

3. The first case should have a break statement while the default case does not need to.



```
Go Run Terminal Help le.c - 1ST LE
C le.c X
C le.c > main(void)
1 #include <stdio.h>
2 int main(void){
3     int n;
4
5     switch (n){
6         case 1:
7             printf("The number is 1");
8             break;
9         case 2:
10            printf("The number is 2");
11            break;
12        default:
13            printf("The number is not 1 or 2");
14    }
15
16    return 0;
17 }
```

4. The condition should be (n <= 10) so that 10 is included in the output.



```
Go Run Terminal Help le.c - 1ST LE - Visual Studio
C le.c X
C le.c > main(void)
1 #include <stdio.h>
2 int main(void){
3     int n = 1;
4
5     while (n <= 10){
6         printf("%d\n",n++);
7     }
8     return 0;
9 }
```

Part III. Answer the following questions.

1. When we say uninitialized variable, it is when a variable is declared but is not assigned to a known defined value before it is used. When accessing the value of an uninitialized variable, sometimes it is automatically set to zero so your program could work but when you change to another compiler, it may not work.
2. If no return statement is executed at the end of the main function, then it would be undefined because the control would return to the calling function after the last statement of the function is executed. But if you declared a void return type, then it would still execute without a return statement. Nonetheless, it is still advisable to put a return statement at the end so that your program would be clear.

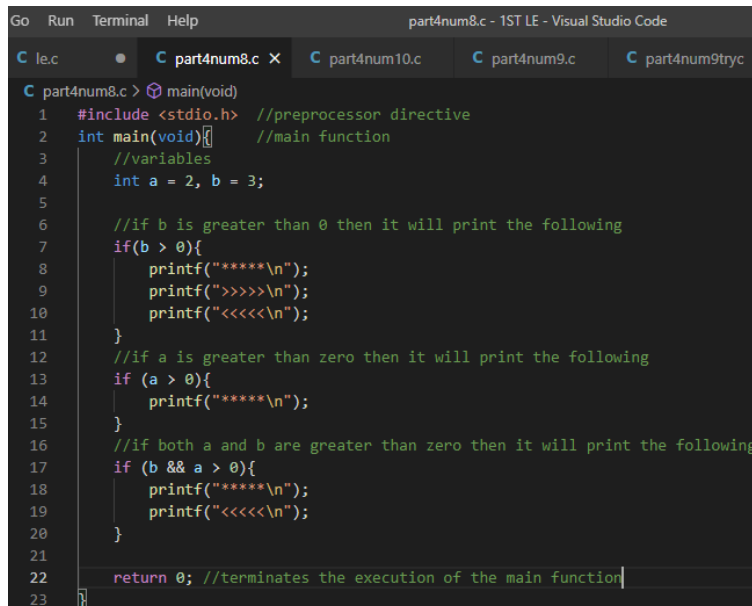
reference: <https://docs.microsoft.com/en-us/cpp/c-language/return-statement-c?view=msvc-170#:~:text=If%20no%20return%20statement%20appears,the%20called%20function%20is%20undefined.>

3. Both are format specifiers wherein %i specify the data type as integer while %d specify the data type as decimal. In print function, the two are almost the same but when it involves scan function, they should be used carefully because it detects base 10 in %d while in %i it auto detects the base.
4. The values of a, c, b are 10, 5.60 and 5 respectively. The value of a changed because it was given that the variable a is an integer type of data and the value (10.3) had a decimal so when it was scanned it, it was then changed to 10 because it only detected the base.
5. The values of a, c, b are 0.00, 0 and 3.89818e-321 respectively. The value of a became 0.00 because initially it was 12.3 (integer type) but when it was scanned it was then changed into a float type. For the value of b, initially it was 789 but when it was scanned it was changed to %g which is a simplifier of the scientific notation float. For the value of c, initially it was 4.5 then it was changed to a decimal type when it was scanned.
6.
  - a.  $(a * b) - (c * d) + e$
  - b.  $((a / b) \% c) / d$
  - c.  $((-a - b) + c) - +d$
  - d.  $((a * -b) / c) - d$
7.
 

```
for (i = 0; j > i; i++)
    j /= 2;
```

#### Part IV. Coding Applications

8.



```

Go Run Terminal Help
part4num8.c - 1ST LE - Visual Studio Code

C le.c C part4num8.c X C part4num10.c C part4num9.c C part4num9tryc

C part4num8.c > main(void)
1 #include <stdio.h> //preprocessor directive
2 int main(void){ //main function
3     //variables
4     int a = 2, b = 3;
5
6     //if b is greater than 0 then it will print the following
7     if(b > 0){
8         printf("*****\n");
9         printf(">>>>\n");
10        printf("<<<<\n");
11    }
12    //if a is greater than zero then it will print the following
13    if (a > 0){
14        printf("*****\n");
15    }
16    //if both a and b are greater than zero then it will print the following
17    if (b && a > 0){
18        printf("*****\n");
19        printf("<<<<\n");
20    }
21
22    return 0; //terminates the execution of the main function
23 }
```

9. reference: <https://www.w3resource.com/c-programming-exercises/basic-declarations-and-expressions/c-programming-basic-exercises-81.php>
10. reference: [https://www.tutorialspoint.com/c\\_standard\\_library/c\\_function\\_fabs.htm](https://www.tutorialspoint.com/c_standard_library/c_function_fabs.htm)

github link: <https://github.com/Maxinne02/Maxinne-Cahilig/upload/main/CMSC21/1st%20E>