1. **DRAW THE TRUTH TABLES** for the following Boolean operations:
   
   (a) **AND**

   (b) **OR**

   (c) **NOT**
2. Write the C Boolean value — either 1, meaning true, or 0, meaning false — that results from computing each of the following C expressions. For parts (d) through (j), **SHOW ALL INTERMEDIATE STEPS.** If you aren’t confident of your answer, write, compile and run a C program to test it.

   (a) ! 1

   (b) 0 && 1

   (c) 1 || 0

   (d) ! 1 || 1

   (e) ! (1 || 1)

   (f) ! (1 && 1)

   (g) ! 1 && 1

   (h) ! 0 && 0

   (i) ! (0 && 0)

   (j) ! 1 || ! 1
3. A C program has the following declarations:

```c
float x = 28.0, y = 36.0, z = 48.0;
int i = 28, j = 40, k = 48;
char sky_is_blue = 1,
char chair_is_green = 0;
char chair_is_brown = 1;
```

**WRITE THE RESULT** of evaluating each of the following expressions. **SHOW ALL INTERMEDIATE STEPS**, including the type of each subexpression (indicating a float with a decimal point). If you aren’t confident about any of your answers, write, compile and run a C program to test it.

(a) sky_is_blue && x <= z

(b) x < y && y < z

(c) k >= j && j >= y

(d) i == x

(e) k != z

(f) chair_is_green || (x + y + z) < 56

(g) chair_is_brown && x + y + k == 56
4. Consider this program:

```c
#include <stdio.h>

int main ()
{ /* main */
    const int constant1 = 10, constant2 = 20,
            constant3 = 14;
    int input_value;
    char current_truth;

    printf("What is the input value?\n");
    scanf("%d", &input_value);
    current_truth = input_value > constant1;
    printf("current_truth = %d\n", current_truth);
    current_truth =
        current_truth && (input_value < constant2);
    printf("current_truth = %d\n", current_truth);
    current_truth =
        current_truth && (input_value == constant3);
    printf("current_truth = %d\n", current_truth);
} /* main */
```

**WHAT IS THE OUTPUT** of this program for the following inputs? If you aren’t confident of your answer, type in, compile and run the program to test it.

(a) 8
(b) 14
(c) 16
(d) 20
5. **ADD A STATEMENT OR STATEMENTS** to the program on the following page (including constant and/or variable declarations if you want) so that the output is the single character `1` followed by a newline. Statements in the execution body of the program must **NOT** include any literal constants (numeric, Boolean or `char`); however, you may declare named constants and/or initialize variables in the declaration section of the program. In the program body, you must use at least **TWO** declared symbolic names (variables or named constants), and you are **ABSOLUTELY FORBIDDEN** to use anything like the following statement:

```c
printf("1\n");
```

On the other hand, you are encouraged to use a `printf` statement that outputs the result of a Boolean expression (which output will be either `1` or `0`).

If you aren’t confident of your answer, type in, compile and run the resulting C program to test it.
#include <stdio.h>

int main ()
{ /* main */
  /*
   ************************************************************
   * Declaration Section
   ************************************************************
   *
   * Named Constants
   */
  const int bits_per_byte = 8;
  const int attention_span_in_seconds = 3;
  /*
   * You can insert stuff after this comment.
   */

  /*
   * Local variables
   */
  int modem_send_speed_in_bits_per_second = 56000;
  int script_file_length_in_bytes = 28000;
  int seconds_to_send_script_file;
  /*
   * You can insert stuff after this comment.
   */

  /*
   ************************************************************
   * Execution Section
   ************************************************************
   *
   * You can insert stuff after this comment.
   */

} /* main */

If you use ANY resources other than Dr. Neeman, the TAs (Badre, Borah, Sadri, Saravanan), the course textbook or the materials posted on the course webpage, you MUST reference them on the quiz. THIS INCLUDES CLASSMATES, FRIENDS, PROFESSORS, ONLINE RESOURCES, ETC.