PROBLEM:

Write a function that begins:

int rotate_left (unsigned num, int n) {

This function should left-shift **num** by **n** positions, where the high-order bits are reintroduced as the low-order bits. Here are two examples of a circular shift operation using a short bit pattern, rather than a full integer.

1000 0001 circular shift 1 yields 0000 001**1**
0110 1011 circular shift 3 yields 0101 1**011**

The main test driver will be provided. No makefile will be required.

Type: **cp -R /gaia/home/faculty/bielr/classfiles_csc60/lab8 .**

Spaces needed: (1) After the **cp** ↑ *Don't miss the space & dot.*
(2) After the **-R**
(3) After the directory name at the end & before the dot.

After the files are in your account and you are still in **csc60**, you need to type: **chmod 755 lab8**
This will give permissions to the directory.
Next move into lab4 directory by typing: **cd lab8**
Type: **chmod 644 lab8.c**
This will give permissions to the file.
Your new lab8 directory should now contain: lab8.c

INPUT/OUTPUT DESCRIPTION:

The input: in a loop, request two unsigned numbers.
The output is printed to the screen by main.

A SAMPLE RUN:

```
Your Name.  Lab 8.

Enter an unsigned integer value (0 to stop):  3

Enter an integer value for the left shift:  4

Original is 3
0000 0000 0000 0000 0000 0000 0000 0011
0000 0000 0000 0000 0000 0000 0011 0000
Shifted it is 48

Enter an unsigned integer value (0 to stop):  5

Enter an integer value for the left shift:  3

Original is 5
0000 0000 0000 0000 0000 0000 0000 0101
0000 0000 0000 0000 0000 0000 0010 1000
Shifted it is 40

Enter an unsigned integer value (0 to stop):  0
```

ALGORITHM DEVELOPMENT - Pseudocode:

```
/*-------------------------------------------------------------*/
main
    do
        print a request and read an integer Number
        if Number is not equal to 0
            print a request and read the number of positions to shift
            print the Original_Number
            print the bit pattern of Original Number
            call rotate_left and return Shifted Number
            print the bit pattern of Shifted Number
            print the Shifted NumberI
         //end if
   while Number is not equal to 0.  //end do-while

/*-------------------------------------------------------------*/
void bitprint (int num)
   find the number of bytes in an unsigned word and change it to number of bits.
   create the mask with a 1 in the left-most position
   for loop thru each bit using count variable
        set the bit to 1 or 0 depending on the result of (num & mask)
        printf the one bit
        if the count is a multiple of four
            print a space
   //end for-loop
   return

/*-------------------------------------------------------------*/
int rotate_left (unsigned num, int n)
   find the number of bytes in an unsigned word and change it to number of bits.
   create the mask with a 1 in the left-most position
   //The Bold represents the code you need to write.
   for loop thru the number-of-bits to shift left
        Save the left bit in variable bit. [This line can be copied from function bitprint.]
        Left shift the num by one
        Add the isolated bit in bit variable onto the right of num
                [This can be done three ways:  (1) +,  (2) | , or  (3) |=   ]
   //end for-loop
   return num

/*-------------------------------------------------------------*/
```

REMINDERS:

Test your program with (3, 4) and (5, 3) as above. Check the validity of your answers.

Prepare Your File For Grading:

When all is well and correct, type: **script StudentName_lab8.txt**

At the prompt, type: **cat lab8.c** to display the code in your session.
type: **gcc lab8.c** to compile the code
type: **a.out** to run the program
type: **7**
type: **4**
type: **7**
type: **8**
type: **0**

After the program run is complete,
type: **exit** to leave the script session

Turn In Completed Session: Go to Canvas and turn in your session (StudentName_lab8.txt).