Lab Assignment #04: Two's Complement

Goal: A stronger understanding of unsigned and signed (2's complement) binary representation.

Objective: To implement a C program that converts a binary string to the corresponding 2's complement

Assignment: (Read the extra credit part before beginning!)

Write a C program that does the following:

1. Read an ASCII string of eight 0's and 1's from the keyboard into an array of chars using the C library function gets.
2. Convert the string into an array of exactly 8 binary digits (integers 0 and 1), so that the least-significant digit is in the first element (subscript 0) of the binary array.

   Note: Use code from your solution to Lab 3.
3. Change the binary number by applying the 2's complement algorithm.
4. Display the resulting binary number, which should be the 2's complement of the original.

   Note: Use code from your solution to Lab 2.

When you run your program, the screen should look something like:

```
Enter a binary string: 11110000
The 2's complement: 00010000
```

Verify that your program operates correctly for binary input strings in the range 00000000 to 11111111, including positive, negative and end-of-range values. Demonstrate your program to the teaching assistant.

5% Extra Credit: (This makes it easier to reuse some of this code in later labs!)

Organize your program into four functions:

1. Function Asc2Bin that was described in Lab 3.
2. A new function with one parameter – an array of eight binary digits (integer 0's or 1's). This function should change the sequence of binary digits by applying the 2's complement algorithm.

   Note: The first element of the binary array (subscript position 0) is the least-significant bit. This function should NOT call printf or scanf.

   Function prototype: `void TwosComp(int binary[8]) ;`
3. Function PrintBin that was described in Lab 2.
4. The main program, which should call the other three functions. The main program should prompt the user for input, get the character string, call the first function to convert the string to an array of binary digits, call the second function to compute the 2's complement, call the third function to compute the 2's complement, and finally call the third function to display the result (with a suitable label).