Assignment 1 (10 points)
Issued 20/2 Due 6/3

Goals: - To practice C programming
- To be familiar with inline Assembly programming within C
- To enforce understanding of doing I/O with and without operating system support

A user is supposed to request all required I/O operations from the operating system. However, in some cases (when?), a user may have to or wish to avoid using the operating system to perform I/O operations. In such cases, the user may try to talk to the hardware directly or easier, to use the BIOS services. In this assignment, we will practice using the BIOS to do some simple I/O operations.

Task 0:

We need first to set up the environment that we will use to develop the programs in this assignment and next ones. To compile/test you program, you will need to install a DOS virtual environment. Install DOSBox (free download from http://www.dosbox.com), which emulates an IBM PC compatible computer running the older operating system, MS-DOS. Then install a C/C++ compiler for DOS, e.g., legacy Turbo or Borland C++ (you will need to install it from within DOSBox). You may find an installer to Turbo C++ and DOXBox on the Resources page of the course website that will help to do both steps at the same time. Experiment and be familiar with this environment.

Task 1:

Write a C program to display a simple message on the screen. You may use the C function **`puts (char *)`**. Then try to use the BIOS interrupt 10H to do the same thing. In particular, we will write our own function, **`biosputs(char *)`**, that will display a string on the screen. Below is one way to write this function using inline Assembly from within a C program.

```
void biosputs (char * str)
 {     int i=0; char
holdc;
 while (str[i]) {
       holdc= str[i++];
       _asm { mov ah, 0Eh
           mov al, holdc
           mov bx, 7 int 10h
         }   // end asm
     } // end while
 } // end function
```

Write a C program to test the above function. Note there are other sub-functions from int 10H to do the same thing (?).

Task 2:

Write a C program to read a string from the keyboard then display it again on the screen. At first, you may use the C function **`gets (char *)`** to get the input from the keyboard. Then write your own **`biosgets (char *)`** function that uses the BIOS to do the same task. Write a C program to test your function.