CS 135 - Computer Science I

Design Assignment 7 (DA7-11/08)

Programming Assignment 7 (PA7-11/11)

As specified in your syllabus, you must turn your assignments in by 6:00 pm on the due date specified. If it is turned in late, but prior to 12:00 midnight the day it is due, credit will be reduced by 50% of the earned score. Any laboratories turned in more than 6 hours late will not earn any credit.

Objectives:

1) You will use a design procedure to develop a well-organized and modular program
2) You will use reference parameters to develop functions with multiple return quantities
3) You will use file output (I/O) to access and process data from a file
4) You will use simple one-dimensional arrays to store and manipulate data
5) You will use simple loops to download or write integer values to a file
6) You will use a functions to conduct various array and file operations

Tasks:

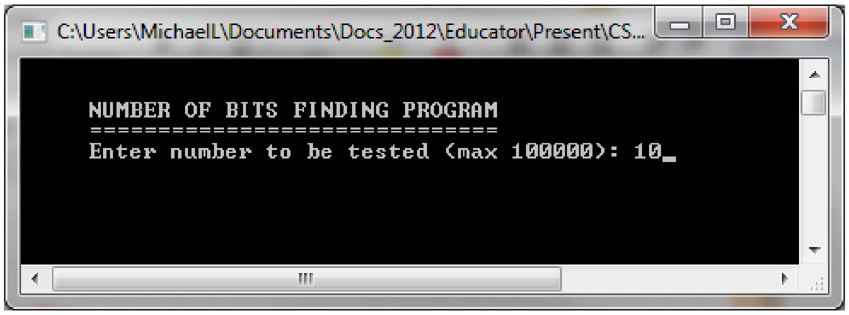
Solving a Little BITty Problem

1) Your program name will be **`numbits.cpp`**. You must use only **`iostream`** I/O operations such as **`cout`** or **`cin`** for the console I/O in this program.

2) You will also need to download the example program so you can be provided examples of test conditions.

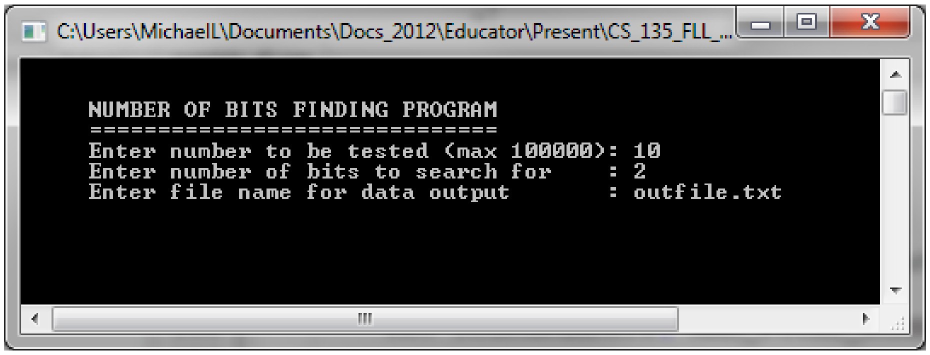
3) You will be writing a program that solves an interesting problem. You will pick a number, say 10, and specify a number of one-bits, say 2, for all the binary values between one and the given number, and this program will tell you how many numbers there are. For these data, the binary numbers between one and ten are 0001, 0010, **0011**, 0100, **0101**, **0110**, 0111, 1000, **1001**, **1010**. There are five values (shown in enlarged bold) between one and 10 that have two one-bit values in the number. Your program will conduct that analysis and show the results.

4) The program will first prompt the user for the number to be used, as shown here. The number must be greater than one but less than or equal to 100,000.


C:\Users\MichaelL\Documents\Docs_2012\Educator\Present\CS...

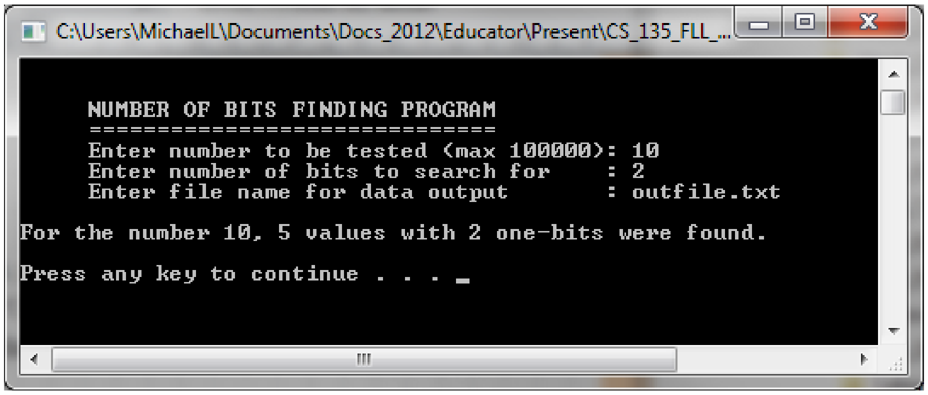
```
    NUMBER OF BITS FINDING PROGRAM
    ==============================
    Enter number to be tested (max 100000): 10_
```

5) If the number is within the specified limits, the program will also prompt for the number of bits to be found and an output file name to which to send result data, as shown here.


C:\Users\MichaelL\Documents\Docs_2012\Educator\Present\CS_135_FLL_...

```
    NUMBER OF BITS FINDING PROGRAM
    ==============================
    Enter number to be tested (max 100000): 10
    Enter number of bits to search for     : 2
    Enter file name for data output        : outfile.txt
```

6) When this data has been entered, the program will provide the results, as shown here.

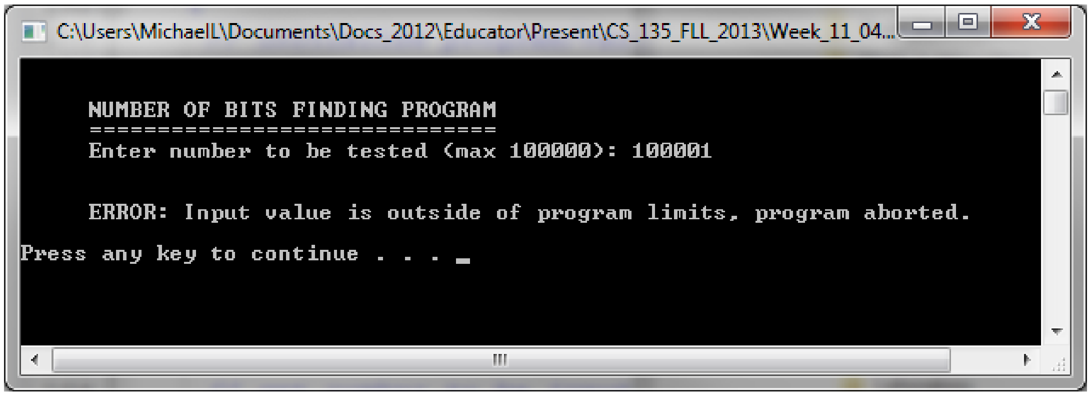

C:\Users\MichaelL\Documents\Docs_2012\Educator\Present\CS_135_FLL_...

```
    NUMBER OF BITS FINDING PROGRAM
    ==============================
    Enter number to be tested (max 100000): 10
    Enter number of bits to search for     : 2
    Enter file name for data output        : outfile.txt

For the number 10, 5 values with 2 one-bits were found.

Press any key to continue . . . _
```

7) If the number is below one or above 100,000, the following error message is displayed. No further processing may be conducted if an error is identified.


C:\Users\MichaelL\Documents\Docs_2012\Educator\Present\CS_135_FLL_2013\Week_11_04...

```
     NUMBER OF BITS FINDING PROGRAM
     ==============================
     Enter number to be tested (max 100000): 100001

     ERROR: Input value is outside of program limits, program aborted.

Press any key to continue . . . _
```

8) In addition to the screen display, a report is printed to the file. Here is an example for the data input provided previously in this document. While it may not be obvious here, there may not be more than 5 values on each display line in the file.

```
Values found up to and including the number 10 that contained
2 one-bits.

The numbers identified are as follows:
3(11), 5(101), 6(110), 9(1001), 10(1010)
```

9) This is not a terribly complicated problem but it will take a few modular steps to make it work. As a rough overview, the process will work as follows:

   a) each number (starting from one up to the input value) is converted to binary; this requires the use of a one-dimensional array to hold the bits (i.e., the complete binary value)

   b) once converted to binary, the number of one-bits is counted in the number

   c) then the number of one-bits is compared to the number of bits entered by the user

   d) if the number of one-bits is the same as the user input, the decimal number is stored in a one-dimensional array, and the one-bit counter is incremented

   e) the max number of bits to manage numbers up to 100,000 will have to be calculated

   f) when it comes time to output the numbers to the file, the function that converts decimal to binary will have to be applied again to display the binary values along with the decimal values

10) Your program should have at least five functions. Be effective with your design.

11) Create a screen shot of each of the following: one screen shot of the input data used in these instructions, and three more using values 1) between 100 and 500, 2) values between 2,000 and 5,000, and 3) values between 50,000 and 75,000. You must specify a different number of one-bits in each program run. Show at least two screen shots for an incorrect input value below 1 and above 100,000.

12) Also provide the output files for the four example programs specified in item #17 above.

Turning in your Design Assignment:

Information:

Week: 11
Laboratory: 11
Design Assignment: 7
Due Date: 11/08, 6:00 pm

To turn in:

The first five steps of the Six Step Programming Process, including:

1. numbits_s1.cpp
2. numbits_s2.cpp
3. numbits_s3.cpp
4. numbits_s4.cpp
5. numbits_s5.cpp

Upload these as separate files, not as a zipped file. Note that following the instructions for uploading your work is critical; you will lose points if you do not follow these instructions. Do not upload any files or in any format other than that specified here.

For information on how to turn in Design Assignments, refer to the "How to Turn in Design Assignments" in the "General Course Information" folder

Turning in your Programming Assignment:

Information:

Week: 10
Laboratory: 10
Programming Assignment: 7
Due Date: 11/11, 6:00 pm

To turn in:

1. The Word file containing the following:

    a. There should be at least six screen shots for the programs as specified in item #17 previously in this document

    b. Remember to clearly annotate every displayed result

2. The executable file:
    a. numbits.exe (numbits_s6.exe is also acceptable)

3. The source code file:
    a. numbits_6.cpp

4. At least <u>four</u> files as specified in item #18 previously in this document:
    a. outfile_1.txt
    b. outfile_2.txt
    c. outfile_3.txt
    d. outfile_4.txt

These files <u>must</u> be compressed and uploaded as one zip file. To do this, select all of the required files, right click on them, and select "Send To", then select "Compressed (zipped) Folder".

Once the folder is created, it will be placed in the same folder in which you are working. Change the name of the zipped folder to "LastnameFirstname_PA0X" (where 'X' is the number of the Programming Assignment) as shown in the following example: "LeveringtonMichael_PA07" (no quotes). After you have renamed the zipped folder, double click on it to verify that it has all the files it is supposed to have.

> Note that following the instructions for uploading your work is critical; you will lose points if you do not follow these instructions. Do not upload any files or in any format other than that specified here.

For information on how to turn in Programming Assignments, refer to the "How to Upload Your Laboratory Assignments" in the "General Course Information" folder