Version: 2/2/2012

ME964
High Performance Computing for Engineering Applications
Assignment 2

Date Assigned: February 2, 2012
Date Due: February 9, 2012

1. There are two related functions that copy/move memory around in C: memcpy and memmove (assume neither of these functions is an alias of the other). When should you use one or the other? Are there any performance issues that should dictate your choice?

2. Write a program that reads a set of words/numbers provided as a command line argument. Pass them to a function that you write. Within the function count the total number of characters in the input excluding white spaces. Return this value and print it out in the main program. Note: the set of words/numbers can contain combinations of letters and digits. Don't use other special characters. Here's an possible scenario:

```
>> myProgram this is a test 4 you 2
```

The output of the program should be 16, since there are 16 characters in the input provided above (excluding white spaces). For this problem you might want to use the variables `argv` and `argc` that are available to you in the main function of your program.

3. Write a C program that takes as an argument an input file, it opens the file, reads all the integer numbers (no more than 1,000,000 of them) and sorts them from the smallest to the largest. The input file contains one integer per line.

   a. Without consulting any online source or book, use an algorithm that you are familiar with to generate a program that prints out four things: (*i*) the number of integers read in, (*ii*) the smallest integer read in, (*iii*) the largest one, and (*iv*) the amount of time in milliseconds (ms) required to perform a sort on the array of integers.
   b. In order to gauge the performance of your sort algorithm consult a book, online source, or use a library that has a sort function implemented to sort the same array as at a) above to verify your answer.
   c. To understand the behavior of the two approaches at a) and b) above, run them on a set of nine input files containing 2^{10}, 2^{11},…, 2^{19} integers generated randomly. Record the time in ms and run a standard regression to approximate the curves that give ordering time versus number of integers for the two algorithms. Provide the plot in png format and indicate the regression function.

4. This problem is identical to Problem 3, except that instead of the sorting operation you have to perform an exclusive scan operation. For a definition of the scan operation

read pages 1 through 3 of the Blelloch 1990 paper available on the class website. Your program should be able to take one input argument which is the name of the text file storing the integers. The input file contains one integer per line.

a. Implement an algorithm that executes an *exclusive scan* operation and prints out three things: (*i*) the number of integers read in, (*ii*) the last entry in the scanned array, and (*iii*) the amount of time in milliseconds (ms) required to perform the scan operation.
b. Run an order analysis much like you did in for 3.c above this time only for your code. Test your code with an input file with 2^5, 2^6,…, 2^{12} integers between -10 and 10 randomly generated. Produce the plot and provide the regression function.

FURTHER NOTES ON YOUR HOMEWORK:

- It doesn't really matter how slow your sorting algorithm runs in Problems 3 or 4. The point there is to understand how difficult it is to design and implement a fast sorting algorithm
- If your code is so fast that it basically report 0 ms, simply include that value in the plot you were asked to generate
- When you generate your plots you don't have to do that within your program. Simply use MATLAB of some python utility to this end
- Please make sure your directory contains a `readme.txt` file that contains any text that you want the TA to read
- You are encouraged to use the ME964 Forum for exchanging ideas or sharing links to helpful information, but no posting of the solutions please.