CS 251, Spring 2009
Voicu Popescu
Assignment 9
Due Friday April 10, 11:59PM

Sorting

1. Implement **void quicksort**(unsigned int *a*, int *n*) as a "C" style function in the file quicksort.cc

    a. The function sorts the array *a* of *n* positive integers with the quick sort algorithm.

    b. The pivot should always be chosen as the first element of the array.

    c. The main function in quicksort.cc reads from the file "quicksortinput" the *n* integers and writes the *n* integers to the file "quicksortoutput" in ascending sorted order. See "quicksortsampleinput" and "quicksortsampleoutput" for formatting details. Please make sure that your program can EXACTLY reproduce that output file given the input file. You may want to use the `diff' program to test this.
    IMPORTANT: Note that the input and output files do not have txt extension.

    d. Run your function on 10 pseudo-randomly generated arrays of integers with lengths 1,000, 2,000, …, and 10,000 (i.e. one thousand through ten thousand). **Count the actual number of array element comparisons and graph it as a function of *n*. Measure the actual running time and graph it as a function of *n*.** Titles, axes labels and units are required. Place both graphs in the file "quicksort.pdf".

    e. See the given quicksort.cc file for how to determine running times. You can modify this file as you wish but I should be able to run it to get your output in the file specified.

2. Implement **void radixsort**(unsigned int *a*, int *n*) as a "C" style function in the file radixsort.cc

    a. The function sorts the array *a* of *n* positive integers using the radix sort algorithm.

    b. The elements of the array are positive and are represented with 32 bits.

    c. The main function in radixsort.cc reads from the file "radixsortinput" the *n* integers and writes the *n* integers to the file "radixsortoutput" in ascending sorted order. See "radixsortsampleinput" and "radixsortsampleoutput" for formatting details. Please make sure that your program can EXACTLY reproduce that output file given the input file. You may want to use the `diff' program to test this.
    IMPORTANT: Note that the input and output files do not have txt extension.

    d. Run your function on 10 pseudo-randomly generated arrays of integers with lengths 1,000, 2,000, …, and 10,000 (i.e. one thousand through ten thousand). **Count the actual number of array element to bucket assignments and plot it in a graph as a function of *n*. Measure the actual running time and graph it as a function of *n*.** Titles, axes labels, and units are required. Place both graphs in the file "radixsort.pdf".

    e. See the given radixsort.cc file for how to determine running times. You can modify this file as you wish but I should be able to run it to get your output in the file specified.

3. Turn in instructions

a. Turn in files: quicksort.cc, quicksort.pdf, radixsort.cc, radixsort.pdf
IMPORTANT: Please be careful with the file names, i.e. capitalization, extensions, etc.
b. A Makefile is not required.
c. DO NOT change any of the function signatures. Doing so will most likely lead to a compile error and a 0 for your overall program grade.
d. Do not partition your code into files we haven't asked for. **IMPORTANT:** Any files other than quicksort.cc, quicksort.pdf, radixsort.cc, radixsort.pdf WILL NOT BE GRADED!
e. Your codes MUST compile on lore.cs.purdue.edu for it to be graded with /opt/csw/gcc3/bin/g++
f. Your code is **REQUIRED** to compile with /opt/csw/gcc3/bin/g++ (gcc version 3.4.5) on lore.cs.purdue.edu for it to be graded. If your code doesn't compile with this compiler, then even if it compiles with another, your code will still be considered noncompiling.
g. All of your documents (quicksort.cc, quicksort.pdf, radixsort.cc, radixsort.pdf) MUST be in the directory A9. In the directory above A9, run the command on lore.cs.purdue.edu

```
turnin -c cs251 -p A9 A9
```

optionally run the following command to verify what files were submitted

```
turnin -c cs251 -p A9 –v
```

Warning: turnin will be automatically disabled at 04/10/2009 at 11:59 PM EST and you will be unable to turnin your assignment after this time.

Warning: turning in multiple times is ok but only the last one will be graded as each turnin permanently overwrites the previous one.

Warning: failure to turnin exactly according to the instructions given above will result in your A9 receiving a grade of 0.