Algorithms Homework Assignment #5

"MSD Radix Sort" (3+1 points)

Due date: Tuesday, May 23rd 2018, 9AM
- Submit softcopy of your source code on the server. (Optional: submit report during class)

In this homework assignment, you'll implement the "MSD Radix Sort" algorithms, and measure the running time. Optionally, plot a graph that compares the running times of "MSD Radix Sort" and other sorting algorithms.

Here are the steps and requirements on what you'll need to do.

Download the text file "hw-sort-int-input-*.txt". This file contains '**K**' integers in random order. You should use this file as input to test your program. However, your program should work with other input files also.

Your task is to simply sort the first 'N' numbers in this file using "**MSD Radix Sort**" algorithms, and measure the running time. Specifically, your program should do the following;

- For a given N value, read the first N *unsigned* integers from the input file, put them into an array, and sort them using "MSD Radix Sort" sorting algorithm.
  - if N > K, then your program should sort all K numbers in the file correctly.
- You may assume that all numbers in the file are unsigned integers (never negative).
- You may assume that all numbers fit within 32-bit unsigned integer (never over 2^{32}-1).
- You must sort in *ascending order* of the numbers (smaller → larger).
- Your program must output the sorted result as well as the running time of the program in milliseconds.

```
$ ./msd_radix_sort  hw-sort-int-input-1.txt  1000000
1
2
3
...
999999
1000000
Running time = xxxxxx ms
$
```

You must implement using **C**, and you must implement a program "msd_radix_sort.c". The file names must be exactly as given, otherwise it won't be graded and you will get zero points.

Compile your programming using gcc on linux

- Ex> `gcc -Wall -o <executable file> <source file>`
- Ex> `gcc -Wall -o msd_radix_sort msd_radix_sort.c`

Your program must take two command line arguments: <input filename> and <N>

- Ex> `./<your_program> <input_file> <N>`
- Ex> `./msd_radix_sort hw-sort-int-input-1.txt 1000000`
- Ex> `./msd_radix_sort some_other_input.txt 10000000`

To measure running time, you may use

- the 'gettimeofday()' function or the 'clock()' function in your C program.
- Please be careful about measuring time for small N values; the time will be very small, you need to think carefully about how to measure that small amount of time.

Warning

- If the performance is not what you expected, you should think carefully about what might be wrong.
- You program must work with other input files as well, not just 'hw-sort-int-input-1.txt'. We will test and grade your program with other input files with different number of integers. Also, your program must gracefully handle all exception/error cases such as N > K, N < K, no input file, incorrect command line arguments, empty lines in the input file (especially the last line), and different EOL formats.

What and how to submit

- You must submit your source code.
- Here is the instruction on how to submit the softcopy files :
  ① Login to your server account at nsl2.cau.ac.kr.
  ② In your home directory, create a directory "submit_alg/submit_<student ID>_hw5" (ex> "/student/20169999/submit_alg/submit_20169999_hw5")
  ③ Put your "msd_radix_sort.c" file in that directory
- Your submission will not be graded if you do not follow the instructions exactly.

Other requirements:

- Your program must run on our class Linux server at nsl2.cau.ac.kr.
- Your code must include your name and student ID at the beginning of the code as a comment.
- Your code should be easily readable and include sufficient comments for easy understanding.
- Your code must be properly indented. Bad indentation/formatting will result in score deduction.
- Your code should not include any Korean characters.
- Your program should work regardless of whether the input file format is Windows/Linux/MacOS based.
  - That is, input file may have Windows or Linux or MacOS line ending format (\n, \r\n, \r).

Grading criteria:

- You get 3 points if your program works as required
  ✓ AND if you meet all of other requirements, AND if your program is fast enough! (think!)
- Otherwise, partial deduction may apply.
- No delayed submissions are accepted.
- Copying other student's work will result in negative points.
- Code that does not compile or code that does not run will result in negative points.
- You may get optional extra credit of up to 1 point if you do the optional extra credit task below.

[Optional] Extra credit task: (up to +1 points)

You should already have programs for "Selection Sort", "Insertion Sort", "Merge Sort", and "Quick Sort". We want to compare the running time of MSD Radix Sort with all other sorting algorithms.

Run your experiment with at least N = 1000, 10000, 100000, 200000, 500000, <student ID>%1000000, 1000000, (possibly more) and measure the running time for all the sorting algorithms that you've implemented so far.

Plot two graphs that compares the running time of the three sorting algorithms. (1 point)

- One with N on the x-axis, running time on the y-axis (in milliseconds)
- Another one with log(N) on the x-axis, log(running time) on the y-axis
- Each graph must plot all sorting algorithms.

What and how to submit

- You must submit a **hardcopy report document report,** as well as a softcopy file of your report.
  - This report document should contain the two graphs that compare the running times of all the sorting algorithms. Do not write code in the document.
- Submit softcopy of the report in the same directory as above. (prefer .docx file, although .hwp is okay)
- Submit hardcopy of your report document during class.