Indian Institute of Technology Bombay
Department of Electrical Engineering
EE-717 Advanced Computing for Electrical Engineers

Assignment 2 (Part 2)

Submission Deadline: September 3, 2015 (Thursday), 11:55 pm (IST)

Note: Part 1 of the assignment is to be done on the platform and the deadline for that is one day ahead of this (September 2, 2015 11:55 pm)

Start Part 2 only after completing Part 1. Note that Part 2 will take non-trivial time. Plan your work accordingly.

The goal of this assignment is to measure runtime and observe the growth of runtime with input size. In this assignment, you have to write a program to sort numbers using *Insertion Sort* ***and*** *Heap Sort.* You will also measure the performance and compare the two methods.

In the subsequent sections, we assume that you are using a Unix based environment. The following commands were tested in Ubuntu 14.04 LTS. However, any compiling environment with standard libraries installed should be ok.

1. Download

Download “assignment2.tar.gz” from the moodle page. Open a terminal. Use the following command at the command prompt ($) to uncompress the file:

```
$ tar xvf assignment2.tar.gz
```

Note that anything that follows the “$” sign should be typed on the command prompt of the terminal.

This should create a directory called *assignment2.* Move to the directory by typing

```
$ cd assignment2
```

2. What is in the directory?

You will see two .c files namely insertion_template.c and heapsort_template.c. You will also see 12 input files named input<num>.txt. These are the input files on which you have to test your programs and measure runtime.

Run

```
        $ make insertionSort
```

to build insertionSort executable and

```
        $ make heapSort
```

to build heapSort executable.

There is a file called *Makefile* which is used by programmers on Unix like platforms to automate compilation of large projects. Any edit that you do to the source file insertion_template.cpp or heapSort_template.cpp must be followed by running the appropriate *make <name>* on the command line to create a new executable.

As of now, the executables do not do any work. Once you are done with the submission on the platform, cut and paste the appropriate code into the function insertionSort() and into functions maxHeapify(), extractMax() and buildHeap(). Specifically, do not overwrite the main functions in the template. This part of the experiment assumes that your programs are correct. Therefore the main() function does not print the outputs and the template does not provide print<name> () functions.

There are also two files runHeapsort.sh and runInsertionsort.sh which are scripts to run the executables on all inputs.

3. Experiments

There are 12 sample input files provided, with inputs of size varying from 100 to 1,000,000. The format for the file is a single line containing *N (*the number of inputs) followed by N integers.

Run insertion sort and heap sort on all inputs by typing

```
        $ ./runInsertion.sh
        $ ./runHeapsort.sh
```

You will notice that the script will run the executables with all the inputs and will print the time taken to run the executable in ms. **Note that the executables should be filled up with your actual code. Running the script on the empty**

template is not going to give meaningful results.

Note down the runtime for each input size vs time in ms.

1. We expect Insertion Sort to take O(n^2) time. Plot the execution time on a spread sheet to see if this is indeed the case.
2. We expect Heap Sort to take O(n log n) time. Plot the execution time on a spread sheet to see if this is the case.

Once you are satisfied that the growth is as expected, do the following:

1. For the trend in runtimes of insertion sort fit a quadratic equation that predicts runtime of an input instance of size n. Let us assume that the quadratic function is of the form ax^2+bx+c. Find best values of a, b and c that explain the behavior. Since the inputs are random, this experiment should capture the average case behavior. Note down a, b and c.
2. For the trend in runtimes of heap sort fit the equation a*n*(lg n) + b where lg n is logarithm base 2. Find best values of a and b that explain the behavior. Since the inputs are random, this experiment should capture the average case behavior. Note down a and b.

4. What to submit?

A google form is set up at http://goo.gl/forms/T7CRQA2IaZ where you can put in the input size, runtimes and the coefficients. Fill it up before the deadline. You can make changes to the form after your initial submisison. You will have to login into your gmail account for the submission of google form.