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

Assignment 1

Submission Deadline: August 10, 2014 (Sunday), 11:30 pm (IST)

Note: You can use either C or C++ to do this assignment. You must follow academic ethics. Plagiarism is punishable. Any academic unethical act may result in a FAIL grade.

The goal of this assignment is to get used to the programming framework and to understand the requirements for programming assignment submissions. In this assignment, you have to write a program to sort numbers using *Insertion Sort* and measure the performance.

In the subsequent sections, we assume that you are using a Unix based environment. The following commands were tested in Ubuntu 14.04 LTS.

1. Download

Download “assignment1.tar.gz” from the moodle page. Open a terminal. Use the following command at the command prompt ($) to uncompress the file:

```
$tar xvf assignment1.tar.gz
```

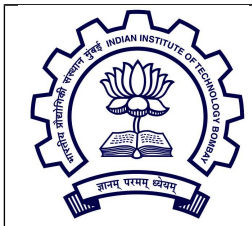
This should create a directory called *assignment1.* Move to the directory and run *make.*

```
$ cd assignment1
$ make
```

This should create an executable called *insertion_sort.* You can run the executable by typing

```
$ ./insertion_sort
```

This executable does not do anything for now. Your homework is to write a program to perform insertion sort.



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

2. What is in the directory?

You are given a template C++ program called *insertion_sort.cpp.* The program has a function called insertionSort() which has code to read a file and write results to a file. You have to fill in the code that does insertion sort.

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 *insetion_sort.cpp* must be followed by running *make* on the command line to create a new executable.

3. Experiments

There are three sample input files provided, with inputs of size 100, 1000 and 10000. The format for the file is a single line containing *N (*the number of inputs) followed by N lines each containing a single integer. Your program must write to an output file containing the runtime in the first line(as printed by the sample code that we have given) followed by inputs sorted in ascending order (one in each line). *Note that your program may fail for inputs which are very very large because you may run out of memory.*

1. Write the portion of the code that is missing.
2. For each of the input files, change the source code to read in the appropriate input file and run your executable. This should create output files with names that you have chosen.
3. Once you are sure that your code is indeed working for the test cases provided, look at the trend in runtimes and fit a quadratic function to the runtimes for the three cases. 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.
4. Take the sorted output and supply those as inputs once more . If you are supplying the output files from step 3, remember to remove the first line (which contains the runtime) and replace it with appropriate N. Since the inputs are already sorted, this should capture the best case behavior of insertion sort. The run time must be of the form mx+n. Note down m and n.
5. Change your input program to sort in descending order. Take the original input files given to you, use your new program to sort in descending order and sort this in ascending order. Now you are exercising the worst case behavior of insertion sort. The runtime must be of the form px^2+qx+r. Note down p, q and r.

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

4. What to submit?

You should upload a single .zip file containing the following files in the specified directory structure.

1. The filename must be <rollno>.zip where <rollno> is your roll number. The roll number must be in lower case only.
2. If we unzip the file, it should contain the following files in the specified directory structure:
    1. A directory with your roll number as its name
    2. Under the directory, the makefile that we supplied and the program that sorts input files in ascending order using insertion sort.
    3. A single file named “README.txt” containing
        1. Your roll number and name in the first line
        2. Run times for average case, best case and worst case for 100 inputs, all printed in one line
        3. Run times for average case, best case and worst case for 1000 inputs, all printed in one line
        4. Run times for average case, best case and worst case for 10000 inputs, all printed in one line
        5. The values of a, b and c, all printed in one line
        6. The values of m and n, all printed in one line
        7. The values of p, q and r, all printed in one line

    If there is more than one field in a single line, separate the fields by a single white space.

Your .zip file should contain these files ONLY and NOTHING ELSE. Specifically, **do not include** the input files, output files and your executable. Follow the guidelines carefully.

5. How to submit?

You will have to submit the .zip file on Moodle by 11:30 pm on Sunday 10th August 2014. You can upload multiple times but the last version you upload will be considered for grading. No extensions for time will be granted.