Indian Institute of Technology Mandi
February-June 2015 Semester

CS202: Advanced Data Structure and Algorithms

Programming Assignment 3

Last date of submission: 29rd March, 2016

1. Implement the following non-comparison sorting algorithms using C++ programing language and sort the input sequence in **ascending order**.
   a) Count sort
   b) 3 versions of Bucket/Bin sort: One version using **linked list** data structure, second version using **stack** data structure and third version using **queue** data structure

   Program should take *n* inputs given from the user and display the sorted sequence.
2. Datasets with different input sizes are given. **Use only the unsorded datasets from Assignment-1**. Implement the above mentioned sorting algorithms by reading the inputs from the file. Observe the running time for the above mentioned sorting algorithms for each of the data with different input sizes. Note down the running time for the dataset and compare the running time for each of the algorithms by plotting the recorded running times for each dataset for each algorithms. Also plot the graph showing the asymptotic running time for the different algorithms as a functon of input size (use the different input sizes provided to you). Store the graph as an image file.

Note:

1. The program should have both the options of reading the numbers to be sorted from user as well as reading them from file.
2. Each array is implemented using the vector data structure (vector.hpp) implemented in the Assignment 1.
3. Update your sorter.hpp classes used in the Assignment 2 as given in the provided sample sorter.hpp to include Count sort and Bucket sort.
4. Implement 3 versions of Bucket sort. One using linked list data structure (list.hpp is provided), second one using stack data structure (stack.hpp is given) and third one using queue data structure (queue.hpp is given)
5. Everyone should use the vector.hpp, sorter.hpp, stack.hpp and queue.hpp classes provided. Do not change the class names. It is expected to strictly use the interfaces provided in the classes to implement the tasks.
6. Submit the graph along with the codes in a single zip file.