Birla Institute of Technology & Science, Pilani
2nd Semester 2016-17 - CS F211 – Data Structures and Algorithms

Lab 6 – 21st Jan. 2017
Topics – Bucket / Bin Sorting and Variants

1. Multi-key Sorting of Sparse Lists:
    - Consider a list Ls of points in 2-dimensional integer grid i.e. points represented as pairs <x,y> of integers:
        - Assume that the points are unique.
        - Also assume the range of the dimensions of the grid are known i.e. for any point <x,y> it is known that xLo <= x <= xHi, and yLo<= y <= yHi.
    - The list Ls is said to be sparse if the size of Ls is O(m+n) where m=xHi-Xlo and n=yHi-yLo; Ls is said to be dense otherwise.
    - Sorting of such a list Ls of Points can be done by bucketing based on the x-value and then inserting into a sorted list which is sorted based on the y-value (or alternatively by bucketing based on the y-value and then inserting into a sorted list that is sorted based on the x-value).
    - For e.g. a point <3,17>
        - is inserted – in the right order – into a linked list that is the x-bucket indexed 3.
        - or alternatively, is inserted - in the right order – into a linked list that is the y-bucket indexed 17.
    - Implement this sorting scheme using an array of linked lists.
    - The sorted list must be copied back into Ls.
    - Note that the sorting algorithm should accept the ranges (i.e. xLo,xHi,yLo,yHi) as parameters and allocate the array dynamically. Bucketing is done on the basis of x-values if the x-range is larger than the y-range (and vice-versa). [Hint: This leaves the insertions in each bucket to be done on shorter lists.]

Parameterize your code based on the following functions.

Key	Function	Input Format	Description
0	readData	0 M xLo xHi yLo yHi	Indicates that next M lines will contain data to be read. Each line will contain a tuple <X Y> separated by space. You must create a 2-dimensional array that stores these read points. xLo, xHi, yLo and yHi are the values as explained before.
1	SortSparseLists	1	Sorts the sparse list as explained by the above algorithm and writes back the sorted tuples to the 2-dimensional array that contained the input read. Also, this function must print the sorted list.

2. Multi-key Sorting of Dense Lists:
    a. If Ls is dense, the points in x-bucket can in turn be bucketed, i.e. a point <3,17> in a space ranging from <0,0> to <40,100> for instance, is stored in a y-bucket indexed 17 within a x-bucket indexed 3.
    b. Implement this sorting scheme using an array of arrays for buckets.
    c. The sorted list must be copied back into Ls.

d. Note that the sorting algorithm should accept the ranges (i.e. xLo,xHi,yLo,yHi) as parameters and allocate arrays dynamically.

Use the above code, and create an additional function as per the following table.

Key	Function	Input Format	Description
2	SortDenseLists	2	Sorts the dense list as explained by the above algorithm and writes back the sorted tuples to the 2-dimensional array that contained the input read. Also, this function must print the sorted list.

3. Instrument the two procedures (i.e. your solutions to 1 and 3) to measure the time taken and heap space used:
    a. Measure the time taken and space used by the two procedures for different input lists (some of which are

        sparse, i.e. of size from O(k) for k k<<m and k<<n to size c*(m+n) for a small constant c, where m and n are as defined in 1.

        and others are

        dense, i.e. of size from O((m+n)*log(m+n)) to O(m*n)).

    b. Based on these measurements identify ranges of input sizes for which your solution in 1 is better than the other or vice-versa; and ranges where neither is superior.