CO 206: Design and Analysis of Algorithms Autumn 2019

Programming Assignment Arindam Karmakar

Arindam Karmakar

1 Write C/C++/Python code for the following problems:

a) A man has n apples $A = \{A_0, A_1, A_2, ..., A_{n1}\}$ in his bag where A_i is the size of the i-th apple. Suppose that no two apples are of the same size.The man will not tell you the size of any of the A_i's. The man can however, can answer queries of the form IsLargeApples(i,j) which returns $1, -1$ according to whether A_i is bigger than A_j or A_i is smaller than A_i, respectively. Write a program to create an array $S = \{s_0, s_1, s_2, \cdots, s_{n-1}\}$ of indices such that $\{A_{s_0} < A_{s_1} < A_{s_2} < \cdots < A_{s_{n-1}}\}$ is the sorted sequence of the apples in the order of their sizes. Modify merge sort to solve your problem. Your modification should use the array S to compare sizes of the apples. Here you move/swap/copy the indices instead of the sizes. For example, if $s_i = j$, then A_{s_i} is the j-th apple A_j (or the size of the j-th apple). But you cannot access A_j or its size, only IsLargeApples(i,j) can access the array A. However, you are free to make any modification to your own array S.

b) The man also has n packets $P_0, P_1, P_2, \cdots, P_{n1}$. It is given that for every apple A_i there is a unique packet P_j such that A_i fits tightly in P_j. But A_i can also fit in a bigger packet, but does not fit in a smaller packet. Write a program that finds the exact matching of the apples and packets, that is, you need to create an array $M = \{m_0, m_1, m_2, \cdots, m_{n-1}\}$ such that the ball A_i is the exact match for the box P_{m_i}. In Part a), you have already sorted the apples according their sizes. Now if you can sort the packets, you can find the matching. Unfortunately, the man will not tell you anything about the comparisons of packet sizes. Instead he will answer queries of the form TightFit(i,j). The answer is 0 if the packet P_j is the exact match for the ball A_i. If A_i fits in P_j but it is not tight, the answer is 1. The answer is a -1 if apple A_i is bigger than the packetP_j. Create M using modified version pf quick sort and Mergesort. You may or may not use the array S available from Part 1. Now, use TightFit( ) queries, but IsLargeApples( ) queries are not allowed.