Assignment 2
Design and Analysis of Algorithms

Nilanjan Datta

Submission Deadline: 28/02/2023

1. Hrithik is given a data structure D maintaining an extrinsic order on n items, supporting two standard sequence operations: $D.\mathsf{get\_at}(i)$ in worst case $\Theta(1)$ times and $D.\mathsf{set\_at}(i, x)$ in worst case $\Theta(n \log n)$ time. Which comparison based sorting algorithm he should choose to sort the n items?

2. Prabal is working on an embedded device (an ATM) that only has 8 KB of free memory, and he wishes to sort the $2,000,000$ transactions withdrawal history by the amount of money withdrawn (discarding the original order of transactions). Suggest an efficient sorting algorithm for Prabal.

3. Krishnakanta claims that he has designed a Priority Queue in the comparison model with both the following properties: EXTRACT-MAX running in $\Theta(1)$, and BUILD-HEAP running in $\Theta(n)$ time. Justify correctness of his claim.

4. Suppose Sajani places n books on top of one another sorted according to the date of publication. Now Sruti comes and swaps several pairs of adjacent books. Suggest an efficient sorting algorithm to re-sort the books if Sruti performs at most $\log n$ swapping.

5. Mriganka claims that any comparison based sorting algorithm can be made to be stable, without affecting the running time by more than a constant factor. Justify the correctness of Mriganka's statement.

6. Pierre has designed a data structure D supports the following sequence operations:

   $$\mathsf{D.insert\_first(x)}, \mathsf{D.delete\_first()}, \mathsf{D.insert\_last(x)}, \mathsf{D.delete\_last()},$$

   each in $O(1)$ time. In addition, D also supports the operations

   $$\mathsf{D.insert\_at(x,i)}, \mathsf{D.delete\_at(i)},$$

   both of which requires $O(\log n)$ time. Can you device efficient algorithms to implement the following higher level operations using the above lower-level operations:

   (a) $\mathsf{reverse}(D, i, k)$: Reverse in D the order of the k items starting at index i.
   (b) $\mathsf{shift\_left}(D, k)$: Move the first k items in order to the end of the sequene in D.

   Compute the time complexity of each proposed algorithms. Assume that all the delete operations return the value deleted.

7. Let A be an array of n integers containing the numbers $\{1, 2, \ldots, n\}$ in some arbitrary order. For integers i and j such that $1 \leq i < j \leq n$, let $\mathsf{Reverse(A, i, j)}$ be a procedure that reverses the subarray $A[i], A[i+1], \ldots, A[j]$ of the array A while leaving the remaining elements of the array unaffected. Nikhil has suggested the following algorithm to sort the array A:

```
for i := 1 to n-1
    while A[i] != i do
        Reverse(A, i, A[i])
```

Prove correctness of Nikhil's algorithm.