DAVP Week 2 (31-08 to 6-09)

Data Structures for sequences.

[Easy]

1. Describe a data structure for representing sequences so that the following operations on sequences can be performed in time independent of the length of the sequence. Note that the operations can modify the sequences to which they are applied.
a) Push_back(x) : Insert x at the end of the sequence.
b) Reverse() : Reverse the sequence.
c) Concatenate(S) : Insert the sequence S at the end of the sequence to to which the operation is applied. You are allowed to destroy the sequence S.

Ans: Two lists. One reverse

[Medium]

2. A permutation p is a one-to-one and onto function from the set {0,1,..,n-1} to itself. A data structure is required for representing a permutation such that the following operations can be performed in time independent of n.

a) Swap(i,j) : Swap the values of p(i) and p(j).
b) Invert() : Invert the permutation. q is the inverse of p iff q(p(i)) = i for all i.
c) Identity() : Test whether the permutation is an identity, that is, p[i] = i for all i.
d) Involution() : test whether the permutation is an involution, that is, p(p(i)) = i for all i.

Ans: Array P, Array P inverse, count_identity, count_matches (between two arrays)

[Easy]

3. In this problem, we want to maintain a sequence of distinct numbers from {0,...,n-1}. The sequence is initially empty. The operations allowed on the sequence are:

1. Make_last(i) : If i is already present in the sequence then delete all elements that occur after it, otherwise insert i at the end of the sequence. In either case, after this operation, i will be the last element in the sequence.

2. Length() : Returns the length of the sequence.

Describe an implementation such that these operations can be performed in time independent of n.

Soln: Keep 2 arrays a & b, one the original array and another which keeps track of the index of element i. A number that keep track of where the array ends say last. Initialze array b with -1;

```
1. Make_last(i) => if( b[i] == -1 || i> last)//not present in array {a[last+1] = i, b[i] =
   last+1;last++;}
   else // present in array {last = b[i];}
2. Length() => Last
```

[Easy]

a) Given a sequence of numbers, determine whether it is possible to output them in sorted order using a single stack as storage, and if so, find how should it be done.

[Easy]

b) Do the same if you are allowed are single queue as storage. Note that 2, 3, 1, can be sorted with a queue, as after storing 2 and 3, 2 will be deleted first from the queue.

[Easy]

Q2.??

Given a very long singly linked list, and a pointer to a given element. Propose a method to delete a given element without iterating the whole list.

4. There are n counters available for buying a railway ticket, numbered 0 to n-1. A program is required to manage the queues at these counters. Whenever a person arrives, a token is given indicating the counter to which the person should go for buying a ticket. This should be the counter where there are minimum number of persons in the queue at the moment (Ties can be broken arbitrarily). When a person leaves a counter, the token is returned. However, all counters may not be open all the time. So a token can only be assigned to a counter that is open. It can be assumed that all persons assigned a counter when it is open will be served before it closes, and there is at least one open counter at any time.

A data structure is required to implement these operations in time independent of n. The operations are:

a) Open_counter(counter_no)
b) Close_counter(counter_no)
c) counter_no get_token()
d) return_token(counter_no);

5. In this problem, the only data structure available to you is a stack of integers. You are required to read in integers one by one, store them in the stack if necessary, and output the integers in sorted order. This is not always possible. Suppose the numbers are read in
the order 2, 3, 1. Since 2 is read first, it cannot be output immediately,
so it has to be stored in the stack. Next 3 is read which also has to
be stored. Now 1 can be output, but we cannot access 2 from the stack,
since 3 was placed on top of it.

[Medium]
6. Consider the following abstract data type: Its value is a subset of the numbers {0,1,2,...,n-1}. The operations allowed on this are:
a) Insert(i) - Insert element i in the subset if not already present.
b) Delete_all(i) - Delete all elements in the subset that are >= i.
Describe an implementation of this data type so that the time required for an arbitrary sequence of m such operations is O(m). Can you implement it so that the worst-case time is O(1) for both operations?

[Medium]
7. A stack is used to store the command history in a shell. However the history contains the N most recently given commands. Describe an implementation that will allow the user to execute the kth previous command, where k is a user specified number <= N, or a new command. If the kth previous command is re-executed then the commands executed after it are deleted from the history. Describe an implementation that will allow all operations on the history can be performed in O(1) time, independent of N.

8. Consider the same abstract data type as defined in problem 4, but the operations now are:
a) Insert(i) - insert i in the subset if it is not present.
b) Find_missing(i) - find the smallest element which is >= i and is not contained in the subset (if it exists).
Design an efficient implementation for this using the union-find data structure.

Algorithms oriented :

[medium]
1:

Given an array of numbers, for every element say a[i] find k and j such that
k<=i<=j.
a[i] should be the max element in this range
size of the range has to be maximum possible.

1.1 solve if all numbers are distinct
1.2 solve if there are repetitions

Ans: stack se ho jaega

[medium]
Q3: Given a matrix sorted rowwise and columnwise. Give an algo to find a given element in the matrix.

[Hard]
c) The stack number of a sequence is the minimum number of stacks needed,
to print the sequence in sorted order. Find the shortest sequence
with stack number 3. Given a sequence, can you find its stack number?

[Hard]
Given an array of numbers (assume word size to be a constant), preprocess any way you want such that each query GCD(i, j) gives GCD of ai to aj, runs in O(1)
eg. 4 20 40 60 10
GCD(1, 4)=4
GCD(2,5)=10
GCD(1,5)=2
each call to GCD should run in O(1) assuming constant word size.
You get extra marks depending on how fast your preprocessing was.
(O(n^3) gets 1 mark, O(n^2) gets 2 marks, O(nlogn) gets 3 marks)