Data Structures and Algorithms

Programming Assignment #3

Topics : STL

Instructions for Submission

1. Register on spoj.com and a2oj.com. Link your *spoj* account to *a2oj*
2. Go to the front page of *a20j* and search for the contest titled **Crash Course (DS & Algo**. The contest code is **40614**.
3. Register for the contest. You'll see a pop-up with 5 problems. Before submitting the problems, make sure that you're logged into your spoj account. Click on any problem and it should redirect you to the spoj page. Submit your question on **this** page. After submission, your ranking should be updated withing 5 minutes.

Problems

1. **Max - Stack** : In the class, we saw a container called stack which supports insertion and deletion of the latest element in $O(1)$. For this question, you need to design a special stack called **Max - Stack** which can provide the following functionalities

   (a) Insertion of any element in $O(1)$

   (b) A view on the last inserted element in $O(1)$

   (c) Deletion of the last inserted element in $O(1)$

   (d) Number of elements remaining in $O(1)$

   (e) A function to check whether the container is empty in $O(1)$

   (f) *A function to print the maximum of all the elements currently present in the container* in $O(1)$

   Hint : Start out by creating a class called **Max_Stack**. Keep 2 STL stacks as instance variables. Now all the operations can be performed easily.

2. **Pairs with Difference** K : Given an array containing n elements (with possible repetitions), and a number k, find out the number of pairs (a_i, a_j) where $i! = j$ and the difference of a_i and a_j is equal to k.

   Hint : One appraoch is to use binary search 2 times. for each element. The other one is to use maps. Take care of repetitions.

3. **Ada and Friends** : It Should be self explanatory from the problem statement.

   Hint : Use maps and hashing.

4. **Maximum of all Windows of Size K** : A window of length k defined as a contiguous sequence of k elements. In an array of n elements, there can be $n - k$ windows of size k. Your task is to find the maximum element of each of these windows and print them.

   Hint : Come up with an $O(n * k)$ solution and further optimize it to $O(n \log k)$. Finally, use *Double Ended Queue* to do it in $O(n)$. You should only submit the code with $O(n)$ complexity.