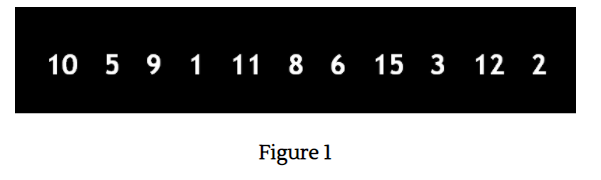
**1. PROBLEM DISCUSSION**

You are given n numbers, representing elements of array a. In case there are two sequences of equal length (and they are also the longest), then print the one for which the starting point of which occurs first in the array. Say, the input array is as shown in figure 1

****

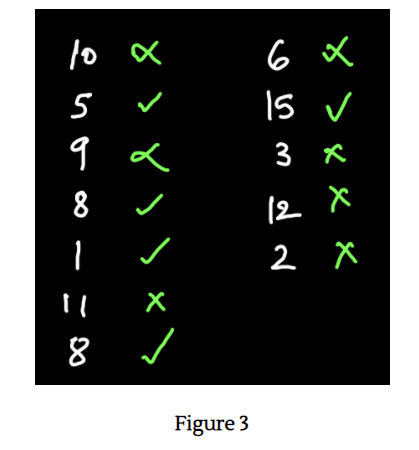
From the above array, the following consecutive sequences can be formed: 1 2 3 5 6 8 9 10 11 12 15 Out of all these sequences, the longest sequence is 8-9-10-11-12 of 5 elements. Hence this should be our output.

**2. Approach :**

We construct a hashmap (Integer vs. Boolean) for all the elements of the array and store "true" against them all initially.

****

This "true" represents that its corresponding element is the starting element of a desired sequence. Now we iterate through the entire loop and check whether a number, 1 less than that element is present in the hashmap or not. If the element is not present then we do nothing. For example, if we are at the element 10 of the array, then we check whether the number previous to 10 i.e. 10-1=9, is present in the hashmap or not. If we find out that the previous number is present in the hashmap, then it could not be the starting of the consecutive sequence. Hence we mark that element as "false". Hence, since 9 is present in the hashmap, therefore 10 is marked as "false". Had the previous number not been present in the hashmap, the current element of the array would have been left as "true". On iterating through all the elements and applying the above algorithm, our hashmap would look like Figure 3.

****

Now, the elements marked as "true" are those elements which are actually at the starting index of a desired consecutive sequence. Now we run a third loop for all the elements and choose those elements which are marked "true". For every "true" element, we find all its consecutive elements in the hashmap. For example, for the element "5", the consecutive sequence becomes 5-6. The size of this sequence is 2. This size is stored as "mh" and the starting element is stored as "mval". Now the next sequence with "1" as the starting element is 1-2-3. The size of the sequence is 3. Since this size is greater than "mh" length, therefore, 3 is the new "mh" and the new "mval" is 1. This goes on until we find the max length out of all the sequences i.e. 5 for the sequence 8-9-10-11-12. Hence "ml" is 5 and "mval" is 8. Now all consecutive elements of the final "mval" till "mh" are printed.

**3. Code**

The code of the above algorithm is fairly straightforward. We want you to write the code on your own first. If you get stuck just refer to the code given below.

ConsoleJava

import java.io.\*;

import java.util.\*;

public class Main {

public static void main(String[] args) throws Exception {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int[]a = new int[n];

for (int i = 0; i < n; i++) {

a[i] = scn.nextInt();

}

HashMap < Integer, Boolean> hm = new HashMap();

for (int val : a) {

hm.put(val, true);

}

for (int val : a) {

if (hm.containsKey(val - 1)) {

hm.put(val, false);

}

}

int mh = 0;

int mval = 0;

for (int val : a) {

if (hm.get(val) == true)

{

int lh = 1;

int lval = val;

while (hm.containsKey(lval + lh)) {

lh++;

}

if (lh > mh) {

mh = lh;

mval = val;

}

}

}

for (int i = 0; i < mh; i++)

{

System.out.println(mval + i);

}

}

}

**4. TIME & SPACE COMPLEXITY**

TIME COMPLEXITY- O(n)

Since the hashmap is traversed once therefore the time complexity is linear.

SPACE COMPLEXITY- O(n) where n= number of entities in the Hashmap.