Part II

4.1.

From the array integer, we find the length of the longest sub-sequence such that elements in the sub-sequence are consecutive integers, the consecutive number can be in any order.

steps

1. Create empty hash
2. Insert all elements into the hash
3. for all elements do the following
4. Check whether element is the starting point of a sub-sequence. This is by checking for arr[i] - 1 in the hash, if not found then it is the first element.
5. if element is first element, then count the number of elements in the consecutive starting with this element. Iterate from arr[i] + 1 till the last element that can be found.
6. If the count is more than the previous longest sub-sequence found, then update this.

In this case only one traversal is needed and the time complexity will be O(n) under the assumption that the hash insert and search takes O(1) time

4.3.

We want to prove that the optimal solution to the big problem contains optimal solution to the sub-problem. We just call prove by induction that this is optimal at each step. Let arr[0..n-1] be the input array and L(i) be the length of the LIS ending at index i such that arr[i] is the last element of the LIS.

5.1

fi = max{ 1

max {Ij+1}

0<j<i, aj<ai

5.2

T(1) = 1

5.3

We define array d[0...n-1], where d[i] is the size n of the sequence ending in element at index i. We will gradually compute this array: d[0], then d[1], and so on. The max value in the array will be the answer to the problem.

let recurred index be i, then

d[i] = max (d[j]+1)

j=0...i-1

a[j]<a[i]

The time complexity is O().

Generate all subsets of size of B[1..n].

For every subset find difference between minimum and maximun elements in it.

Return the minimum difference