19I510 Design and Analysis of Algorithms

**Exercise 3 – Hashing: Collision resolution strategies**

1. Given a string, find the occurrence of each character in the string with the help of hash table. The string may contain any ASCII character (characters, numbers and symbols). Skip the numbers, symbols (or) space which is part of the string. [The logic should treat both 'a' and 'A' as same character]

**Input Format**Input is a string of any length  
**Output Format**The output will print the unique symbols in the string along with occurrence separated by comma

**Sample Input**

GOD IS GREAT

**Sample Output**  
G-2,O-1,D-1,I-1,S-1,R-1,E-1,A-1,T-1



1. Given a set of integers, find the pair with specified sum as x. If there are many pairs with required sum, print the pair with the maximum difference. [use hash map with size equal to the required sum] Hint: Consider only the pairs where first element index is smaller than the index of the second element

**Input Format**The first line of input consists of an integer(N) representing the size of the array.  
The next line consists of N Integers separated by space.

The third line consists of the required sum  
**Output Format**The output will print the pair with required sum x

**Sample Input**

6  
3 10 16 13 5 5

21

**Sample Output**

16,5

1. Smallest subarray with all occurrences of a most frequent element

Given an array, A. Let x be an element in the array. x has the maximum frequency in the array. Find the smallest subsegment of the array which also has x as the maximum frequency element.

**Input** : arr[] = {4, 1, 1, 2, 2, 1, 3, 3}

4 1 1 2 2 1 3 3

**Output** : 1, 1, 2, 2, 1

The most frequent element is 1. The smallest subarray that has all occurrences of it is

1 1 2 2 1

**Input** : A[] = {1, 2, 2, 3, 1}

**Output** : 2, 2

Note that there are two elements that appear two times, 1 and 2. The smallest window for

1 is whole array and smallest window for 2 is {2, 2}. Since window for 2 is smaller, this is

our output.

1. Minimum number of subsets with distinct elements

**You are given an array of n-element. You have to make subsets from the array such that no subset contains duplicate elements. Find out minimum number of subset possible.**

**Input** : arr[] = {1, 2, 3, 4}

**Output** :1

Explanation : A single subset can contains all

values and all values are distinct

**Input** : arr[] = {1, 2, 3, 3}

**Output** : 2

Explanation : We need to create two subsets

{1, 2, 3} and {3} [or {1, 3} and {2, 3}] such

that both subsets have distinct elements.

1. Maximum distance between two occurrences of same element in array.

Given an array with repeated elements, the task is to find the maximum distance between two occurrences of an element.

**Input** : arr[] = {3, 2, 1, 2, 1, 4, 5, 8, 6, 7, 4, 2}

3 2 1 2 1 4 5 8 6 7 4 2

**Output**: 10

// maximum distance for 2 is 11-1 = 10

// maximum distance for 1 is 4-2 = 2

// maximum distance for 4 is 10-5 = 5

1. Minimum array

Given an array with n positive integers. We need to find the minimum number of operations to make all elements equal. We can perform addition, multiplication, subtraction or division with any element on an array element.

**Input** : arr[] = {1, 2, 3, 4}

**Output** : 3

Since all elements are different, we need to perform at least three operations to make them same. For example, we can make them all 1 by doing three subtractions. Or make them all 3 by doing three additions.

**Input** : arr[] = {1, 1, 1, 1}

**Output** : 0

1. Count maximum points on same line.

Given N point on a 2D plane as pair of (x, y) co-ordinates, we need to find maximum number of point which lie on the same line.

**Examples**:

**Input** : points[] = {-1, 1}, {0, 0}, {1, 1},

{2, 2}, {3, 3}, {3, 4}

**Output** : 4

Then maximum number of point which lie on same line are 4, those point are {0, 0}, {1, 1}, {2, 2},

{3, 3}