**Arithematic Progression**

**Description**

In mathematics, an arithmetic progression (AP) or arithmetic sequence is a sequence of numbers such that the difference between the consecutive terms is constant. The difference here means the second minus the first. For instance, sequence 5, 7, 9, 11, 13, 15, . . . is an arithmetic progression with *common difference* of 2.  
Given any three terms of some random Arithematic progression and an integer key, you have to determine if the key will be present in the Arithematic sequence or no. check out sample input/output for more clarification.  
  
----------------------------------------------------------------------  
Input:  
The first line will contain three comma-separated integers representing three consecutive of some Arithmetic Progression  
The second line will have the integer key  
  
Output:  
True if the key is a part of that AP, False otherwise  
  
----------------------------------------------------------------------

Sample input:  
9, 11, 13  
19  
  
Sample output:  
True  
  
----------------------------------------------------------------------  
Sample input:  
9, 11, 13  
20  
  
Sample output:  
False  
  
----------------------------------------------------------------------  
Sample input:  
9, 11, 13  
3  
  
Sample output:  
True

**2) Sort a vector of 0s, 1s, and 2s**

**Description**

Write a code to sort a listof 0s, 1s, and 2s. Note that the time complexity should be O(n).

Note: Do not use the built-in sorted() function.  
  
**Example:  
Input 1:**[0, 2, 0, 0, 1, 1, 2, 0, 1]  
**Output 1:**[0, 0, 0, 0, 1, 1, 1, 2, 2]

#### 3) Diff K

**Description**

You have done the specific sum problem. Now it's time for a similar albeit slightly modified version of the same - **Diff K**.   
  
Given a list 'L' of sorted integers and another non-negative integer k, find if there exist 2 indices i and j such that L[i] - L[j] = k, i != j.  
  
**Format:  
Input:**A single sorted list in the first line and the non-negative integer 'k' in the second line.  
**Output:**'Yes' if there exist two such indices otherwise, 'No'.  
  
**Examples:  
Input 1:**[1, 3, 5, 6, 10]  
4  
**Output 1:**Yes  
  
**Input 2:**[1, 3, 5, 6, 10]  
8  
**Output 2:**No

#### 4) Intersection of Sorted Arrays

**Description**

Find the intersection of two sorted lists, i.e., find the elements that occur in both.  
  
**Format:  
Input:**2 sorted lists  
**Output:**A single sorted list that contains the elements present in both the arrays.  
  
**Examples:  
Input 1:**[1, 2, 3, 5, 9]  
[3, 6, 9, 12]  
**Output 1:**[3, 9]  
  
**Input 2:**[1, 1, 2, 4, 8, 8, 8, 9]  
[1, 2, 6, 7, 8, 8, 8] **Output 2:**[1, 2, 8, 8, 8]

#### 5) Weird Function

**Description**

In data science, quite often you need to implement research papers and write code according to what's present in those papers. Research papers have a lot of maths involved and you need to implement the maths in code. In this exercise, you're required to implement some maths in code. The problem is as follows:  
  
For fixed integers a, b, c, define a weird function F(n) as follows:   
F(n) = n - c for all n > b   
F(n) = F(a + F(a + F(a + F(a + n)))) for all n ≤ b.   
   
Also, define S(a, b, c) = ∑F(n) where n takes the values 0 till b [in other words, S(a, b, c) = F(0) + F(1) + F(2) + .... F(b-1) + F(b)].  
  
The input will be the value of a, b and c. The output should be S(a, b, c). You can define the functions in your own customized way with no restrictions on the number of parameters. For example, you can define the function S which can take additional parameters than a, b and c. Just make sure the code behaves as per the maths.  
   
For example, if a = 20, b = 100 and c = 15, then F(0) = 195 and F(2000) = 1985.   
Therefore, S(20, 100, 15) = 14245  
  
  
Input:  
3 values separated by a comma  
a,b,c  
  
Output:  
S(a,b,c)  
  
  
Sample input:  
20, 100, 15  
  
Sample output:  
14245