/* Problem 1--Perfect Numbers
This was a straightforward iteration of factor sums. You only need to check up to the square root to find all factors. */

import java.io.*;
import java.util.*;

public class prob1 {
  private static Scanner in;
  private static PrintWriter out;
  private static int cs;

  public static void main(String[] args) throws Exception {
    in = new Scanner (new File ("prob1.in"));
    out = new PrintWriter ("prob1.out");
    cs = 1;
    while (true) { //read in the values
      int r = in.nextInt ();
      int k = in.nextInt ();
      int f = in.nextInt ();
      int l = in.nextInt ();
      if (r==0 && k==0 && f==0 && l==0) break;
      Process (f,l,r,k);
    }
    in.close ();
    out.close ();
  }

  //This iterates the Factor Sum r times and checks whether the answer is
  //k times the original number
  public static boolean IsPerfect (int r, int k, int n) throws Exception {
    int p = n;
    for (int i=0; i < r; i++) p = FS (p);
    return p == k*n;
  }

  //Processes each data case
  public static void Process (int f, int l, int r, int k) throws Exception {
    out.print ("Case "+(cs++)+":\r\n\r\n");
    out.print ("The ("+r+","+k+")-perfect numbers between "+f+" and "+l+
    " are:\r\n\n");
    for (int i=f; i <= l; i++)
      if (IsPerfect (r,k,i)) out.print (i+"\r\n");
    out.print ("\r\n");
  }

  //Computes the Factor Sum of n by checking all factors up to the square
  //root.
  public static int FS (int n) throws Exception {
    int sum = 0;
    for (int i = 1; i*i <= n; i++)
      if (n%i==0) {
        //If the number is a perfect square, only count the
        sum += i; //square root once.
        if (n/i > i) sum += n/i;
      }
    return sum;
  }
}
prob1.java

65  }
66