```
 1  /* Problem 2--More Appropriate Numeric Bases
 2     We could solve this with algebra.  However, with such a small number
 3     of bases to check, it's just as easy to trial and error it. */
 4
 5  import java.io.*;
 6  import java.util.*;
 7
 8  public class prob2 {
 9
10   private static Scanner in;
11   private static PrintWriter out;
12   private static int cs;
13
14   public static void main (String[] args) throws Exception {
15
16    in = new Scanner (new File ("prob2.in"));
17    out = new PrintWriter ("prob2.out");
18    cs = 1;
19    while (in.hasNext()) {//read in the factors and the product
20     String m1 = in.next();
21     String m2 = in.next();
22     String p = in.next();
23     Process (m1,m2,p);
24    }
25    in.close ();
26    out.close ();
27   }
28
29   //Process computes the answer for a given input case
30   public static void Process (String m1, String m2, String p)
31       throws Exception{
32
33    out.print ("Case "+(cs++)+": ");
34    int[] b = Bases (m1,m2,p); //find all bases
35    if (b.length==0) //examine cases to make it grammatical
36     out.print (m1+" x "+m2+" never equals "+p);
37    else if (b.length==1)
38     out.print (m1 +" x "+m2 + " equals "+p+" in base "+b[0]);
39    else {
40     out.print (m1+" x "+m2+" equals "+p+" in bases ");
41     for (int i=0; i < b.length; i++)
42      if (i==0) out.print (b[i]); //Oxford comma
43      else if (i==b.length-1 && b.length > 2) out.print (", and "+b[i]);
44      else if (i==b.length-1) out.print (" and "+b[i]);
45      else out.print (", "+b[i]);
46    }
47    out.print (".\r\n\r\n");
48   }
49
50   //Convert the string to a number
51   public static int Convert (String m, int b) throws Exception {
52
53    int n = 0;
54    for (int i=0; i < m.length(); i++)
55     n = n*b+GetValue(m.charAt(i));
56    return n;
57   }
58
59   //Get the value of the character
60   public static int GetValue (char c) throws Exception {
61
62    if (c < 'A') return c-'0';
63    return c-'A'+10;
64   }
```

```
 65
 66   //Get the lowest possible base, a base cannot contain a digit
 67   //greater than or equal to the base
 68   public static int FirstBase (String m1, String m2, String p)
 69    throws Exception {
 70
 71    int fb = 2;
 72    for (int i=0; i < m1.length(); i++) {
 73     int v = GetValue (m1.charAt(i));
 74     if (v >= fb) fb = v+1;
 75    }
 76    for (int i=0; i < m2.length(); i++) {
 77     int v = GetValue (m2.charAt(i));
 78     if (v >= fb) fb = v+1;
 79    }
 80    for (int i=0; i < p.length(); i++) {
 81     int v = GetValue (p.charAt(i));
 82     if (v >= fb) fb = v+1;
 83    }
 84    return fb;
 85   }
 86
 87   //Computes the valid bases and puts them into an array
 88   public static int[] Bases (String m1, String m2, String p)
 89     throws Exception {
 90
 91    int ct = 0; //try all bases in turn
 92    for (int i=FirstBase(m1,m2,p); i <= 36; i++) //does it work?
 93     if (Convert(m1,i)*Convert(m2,i)==Convert(p,i)) ct++;
 94    int[] b = new int[ct];
 95    ct = 0;
 96    for (int i=FirstBase(m1,m2,p); i <= 36; i++) //load up array
 97     if (Convert(m1,i)*Convert(m2,i)==Convert(p,i)) b[ct++] = i;
 98    return b;
 99   }
100  }
101
```