import java.util.*;

public class Project4 {

/*
 * This method converts a number from the decimal numbering system to binary
 * Examples: 3 -> 0011   6 -> 0110   9-> 1001    14 -> 1110
 * Arjun Khanna ACSL junior - contest #3 program
 */

public static int[] decToBinary(int value) {

    //The binary number gets stored into bin
    int[] bin = new int[4];

    //result will get updated as bin gets updated. Starts at value, and subtracts
    the next
    //highest power of 2
    int result = value;

    //digit is the index for bin
    int digit = 0;

    //This loop will go through 4 iterations: i will be 8,4,2,1
    for(int i = 8; i>=1 ; i/=2) {
        //if i is less than the current result, that means that digit is 1
        if(result<i)
            bin[digit] = 0;
        else
            {
            bin[digit] = 1;
            result = result-i;
            }
        digit++;
    }

    return bin;
}
}
/*
* This converts the input strings into a 2x4 array that resembles a Veitch Diagram
* The 'X's will be represented by 1's and empty spaces will be 0's
* Note that this method is assuming that str is 2 characters in length
*/

public static int[][] convertToVeitch(String str)
{
    int[][] veitch = new int[2][4];

    //decValue will get the decimal value of the two characters in str
    int decValue = 0;

    for( int i=0 ; i<str.length() ; i++ )
    {
        //This if statements tells us if the character is 0-9 or A-F
        //Then it converts them into the correct decimal value
        //Ex: 9 -> 9   C -> 12
        if(str.charAt(i)>='0' && str.charAt(i)='<='9')
            decValue = str.charAt(i)-'0';
        else
            decValue = str.charAt(i)-'A'+10;

        //We call the above method to convert to binary, then store it into the final array
        veitch[i] = decToBinary(decValue);
    }
    return veitch;
}

/*
* This method takes care of the first two rules. Grouping by 4 X's.
*
It is done in the specified priorities. Each if statement represents a different case.

If a group of 4 X's is found, all those X's are converted to 0, and str is updated.

Since diagram is an object, the changes in X's will be saved, so only str needs to be returned

```java
public static String RuleOneTwo(String str, int[][] diagram)
{
    if(diagram[0][0]==1 && diagram[0][1]==1 && diagram[0][2]==1 &&
       diagram[0][3]==1)
    {
        //B
        diagram[0][0]=0;
        diagram[0][1]=0;
        diagram[0][2]=0;
        diagram[0][3]=0;
        str+="B +";
    }

    if(diagram[1][0]==1 && diagram[1][1]==1 && diagram[1][2]==1 &&
       diagram[1][3]==1)
    {
        //~B
        diagram[1][0]=0;
        diagram[1][1]=0;
        diagram[1][2]=0;
        diagram[1][3]=0;
        str+="~B +";
    }

    if(diagram[0][0]==1 && diagram[0][1]==1 && diagram[1][0]==1 &&
       diagram[1][1]==1)
    {
        //A
        diagram[0][0]=0;
        diagram[0][1]=0;
    }
```
```c
    diagram[1][0]=0;
    diagram[1][1]=0;
    str+="A +";
}

if(diagram[0][1]==1 && diagram[0][2]==1 && diagram[1][1]==1 &&
    diagram[1][2]==1)
{
    //C
    diagram[0][1]=0;
    diagram[0][2]=0;
    diagram[1][1]=0;
    diagram[1][2]=0;
    str+="C +";
}

if(diagram[0][2]==1 && diagram[0][3]==1 && diagram[1][2]==1 &&
    diagram[1][3]==1)
{
    //~A
    diagram[0][2]=0;
    diagram[0][3]=0;
    diagram[1][2]=0;
    diagram[1][3]=0;
    str+="~A +";
}

if(diagram[0][0]==1 && diagram[1][0]==1 && diagram[0][3]==1 &&
    diagram[1][3]==1)
{
    //~C
    diagram[0][0]=0;
    diagram[1][0]=0;
    diagram[0][3]=0;
    diagram[1][3]=0;
    str+="~C +";
}

return str;
```
/*
   * This method works just like the method RuleOneTwo, just with combinations of 2 X's
   * This will of course have more combinations. If you can figure out a way to do this without
     * a lot of if statements, try it. I would have to think about it for a little longer to
       * figure it out.
   *
   * Remember, all the groups of 2 X's must be then changed to 0, and str must be updated with
     * the correct boolean expression
   */

public static String RuleThreeFour(String str, int[][] diagram)
{
    return str;
}

/*
   * Again, this method is very similar to the above methods, it will need 8 if statements exactly.
   * It just checks each element in the array individually.
   */

public static String RuleFive(String str, int[][] diagram)
{
    return str;
}
public static void main(String[] args) {
    // TODO Auto-generated method stub
    Scanner sc = new Scanner(System.in);

    System.out.println("Enter the 5 lines.");

    //result is our final output, it will contain the boolean expressions
    String result = "";
    int[][] dia = new int[2][4];
    String input1 = sc.nextLine();
    //String input2 = sc.nextLine();
    //String input3 = sc.nextLine();
    //String input4 = sc.nextLine();
    //String input5 = sc.nextLine();

    //converts the input into a 2x4 array
dia=convertToVeitch(input1);

    //performs RuleOneTwo
    result = RuleOneTwo(result,dia);

    //result = RuleThreeFour(result,dia);
    //result = RuleFive(result,dia);

    //The following is just for testing purposes
    System.out.println(result);
    for (int i =0;i<2;i++)
    {
        for(int j=0;j<4;j++)
        {
            System.out.print(dia[i][j]);
            System.out.println();
        }
    }
    sc.close();
}