```
import java.util.*;

public class acsl_C333333 {

	static int [][] aa = {{1, 1, 0, 0}, {1, 1, 0, 0}, {1, 1, 0, 0}, {1, 1, 0, 0}};
	static int [][] bb = {{1, 1, 1, 1}, {1, 1, 1, 1}, {0, 0, 0, 0}, {0, 0, 0, 0}};
	static int [][] cc = {{0, 1, 1, 0}, {0, 1, 1, 0}, {0, 1, 1, 0}, {0, 1, 1, 0}};
	static int [][] dd = {{0, 0, 0, 0}, {1, 1, 1, 1}, {1, 1, 1, 1}, {0, 0, 0, 0}};
	static int [][] notaa = {{0, 0, 1, 1}, {0, 0, 1, 1}, {0, 0, 1, 1}, {0, 0, 1, 1}};
	static int [][] notbb = {{0, 0, 0, 0}, {0, 0, 0, 0}, {1, 1, 1, 1}, {1, 1, 1, 1}};
	static int [][] notcc = {{1, 0, 0, 1}, {1, 0, 0, 1}, {1, 0, 0, 1}, {1, 0, 0, 1}};
	static int [][] notdd = {{1, 1, 1, 1}, {0, 0, 0, 0}, {0, 0, 0, 0}, {1, 1, 1, 1}};

	public static void main (String [] args) {

		Scanner sc = new Scanner (System.in);
		System.out.print("Enter input: ");
		String input = sc.next();

		int [][] board = new int [4][4];
		ArrayList <String> arr = new ArrayList <String> ();

		//split up the input
		int c = 0;
		while(input.indexOf('+') > -1) {
			if (input.charAt(c)=='+') {
				String temp = input.substring(0, c);
				arr.add(temp);
				input = input.substring(c+1);
				c=0;
			}
			c++;
		}
		arr.add(input);

		//reset to 0
		for (int i = 0; i < board.length; i ++) {
			for (int j = 0; j < board.length; j ++) {
```

```
                board[i][j] = 0;
            }
        }

        int [][] merging = new int [4][4];
        //System.out.println("TOTAL: "+arr);
        for(int i = 0; i < arr.size(); i++) {
            String now = arr.get(i); //AB
            //System.out.println("WORKING ON: "+now);
            ArrayList <String> sep = new ArrayList <String> ();
            sep = getSeparate(now); // [A, B]
            //System.out.println(sep);

            for (int j = 0; j < sep.size(); j++) {
                String letter = sep.get(j);
                //System.out.println(letter);
                int [][] letterBoard = getAlpha(letter);
                //System.out.println("BEFORE IF");
                //print(letterBoard);

                if (j == 0) {
                    merging = letterBoard;
                }
                else {
                    merging = merge(merging, letterBoard);
                }
                //System.out.println("AFTER IF");
                //print(merging);
            }

            board = add(board, merging);
        }

        //print(board);

        //System.out.println("ANSWER");
        ArrayList <String> ansArr = new ArrayList <String> ();
        for (int i = 0; i < board.length; i ++) {
```

```
                String ansString = "";

                for (int j = 0; j < board.length; j ++) {
                        ansString += board[i][j];
                }
                ansString = biToHex(ansString);
                ansArr.add(ansString);
        }

        for (int i = 0; i < ansArr.size(); i++) {
                System.out.print(ansArr.get(i));
        }
}

//binary to hexadecimal string
public static String biToHex(String a) {
        int dec = Integer.parseInt(a, 2);
        String hex = Integer.toString(dec, 16);
        return hex;
}

public static int [][] getAlpha (String a) {
        int [][] myArr = new int [4][4];

        if (a.equals("A")) {
                myArr = aa;
        }
        else if (a.equals("~A")) {
                myArr = notaa;
        }
        else if (a.equals("B")) {
                myArr = bb;
        }
        else if (a.equals("~B")) {
                myArr = notbb;
        }
        else if (a.equals("C")) {
                myArr = cc;
```

```
		}
		else if (a.equals("~C")) {
			myArr = notcc;
		}
		else if (a.equals("D")) {
			myArr = dd;
		}
		else if (a.equals("~D")) {
			myArr = notdd;
		}

		return myArr;
}

//Separate out the boolean variables i.e ~AB -> [~A, B]
public static ArrayList<String> getSeparate(String a) {
		ArrayList <String> al = new ArrayList <String> ();
		int y = 0;
		for (int i = 0; i < a.length(); i++) {
			if (a.charAt(i) != '~') {
				al.add(a.substring(i-y, i+1));
				y = 0;
			}
			else {
				y+=1;
				continue;
			}
		}
		return al;
}

//Merging letters (AND Operation)
public static int [][] merge(int[][] grid1, int[][] grid2) {
		int [][] result = new int [4][4];

		for (int i = 0; i < result.length; i ++) {
			for (int j = 0; j < result.length; j ++) {
				if (grid1[i][j] == 1 && grid2[i][j] == 1) {
```

```
                                        result[i][j] = 1;
                                }
                        }
                }
                return result;
        }

        //The +(OR) operation
        public static int [][] add(int[][] grid1, int[][] grid2) {
                //System.out.println("######INSIDE ADD#####");
                int [][] result = new int [4][4];

                for (int i = 0; i < result.length; i ++) {
                        for (int j = 0; j < result.length; j ++) {
                                if (grid1[i][j] == 1 || grid2[i][j] == 1) {
                                        result[i][j] = 1;
                                }
                        }
                }
                //System.out.println("#####OUT OF ADD#####");
                return result;
        }
        //Prints out the 2D array
//      public static void print(int[][] grid)
//      {
//              System.out.println(" ------------------");
//          for (int i = 0; i < grid.length; i++)
//          {
//              for (int j = 0; j < grid[i].length; j++)
//              {
//                      System.out.print(" | ");
//                  System.out.print(grid[i][j]);
//              }
//              System.out.println(" |");
//              System.out.println(" ------------------");
//          }
//      }
    }
```