```
import java.io.File;
import java.util.Scanner;
import java.io.FileNotFoundException;
import java.lang.Math;

public class ACSL_contest3 {
	public static void main(String[] args) {
		String binValue;
		String result;
		String result2;
		String result3;
		try {
			File file = new File("C:\\eclipse-workspace\\MyJavaProj\\src\\input.txt");
			Scanner scanner = new Scanner(file);
			while (scanner.hasNextLine()) {
				String line = scanner.nextLine().trim();
				System.out.println(line);
				binValue = HextoBin(line);
				// System.out.println("From main " + binValue);
				result = group4(binValue);
				// System.out.println("Result is " + result);
				int index = result.indexOf('|');
				if (index != -1) {
					binValue = result.substring(index + 1);
					// System.out.println(" New binValue 1 " + binValue);
					result = result.substring(0, index);
				}
				// System.out.println("Final Result after G4 " + result);
				result2 = group2(binValue);
```

```
                index = -1;
                index = result2.indexOf('|');
                if (index != -1) {
                        binValue = result2.substring(index + 1);
                        // System.out.println(" New binValue 2 " + binValue);
                        if (result.equals(""))
                                result = result2.substring(0, index);
                        else
                                result = result + "+" + result2.substring(0, index);
                }
                result3 = group1(binValue);
                index = -1;
                index = result3.indexOf('|');
                if (index != -1) {
                        binValue = result3.substring(index + 1);
                        // System.out.println(" New binValue 3 " + binValue);
                        if (result.equals(""))
                                result = result3.substring(0, index);
                        else
                                result = result + "+" + result3.substring(0, index);
                }

                System.out.println("Final result is " + result);
        }
} catch (FileNotFoundException e) {
        System.out.println("Error - Expected data file not found");
        e.printStackTrace();
}
```

```
}

private static String HextoBin(String hexdec) {
	StringBuilder retStr = new StringBuilder();
	for (int i = 0; i < hexdec.length(); i++) {
		switch (hexdec.charAt(i)) {
		case '0':
			retStr.append("0000");
			break;
		case '1':
			retStr.append("0001");
			break;
		case '2':
			retStr.append("0010");
			break;
		case '3':
			retStr.append("0011");
			break;
		case '4':
			retStr.append("0100");
			break;
		case '5':
			retStr.append("0101");
			break;
		case '6':
			retStr.append("0110");
			break;
		case '7':
			retStr.append("0111");
```

```
            break;
        case '8':
            retStr.append("1000");
            break;
        case '9':
            retStr.append("1001");
            break;
        case 'A':
        case 'a':
            retStr.append("1010");
            break;
        case 'B':
        case 'b':
            retStr.append("1011");
            break;
        case 'C':
        case 'c':
            retStr.append("1100");
            break;
        case 'D':
        case 'd':
            retStr.append("1101");
            break;
        case 'E':
        case 'e':
            retStr.append("1110");
            break;
        case 'F':
        case 'f':
```

```
                retStr.append("1111");
                break;
            default:
                System.out.print("\nInvalid hexadecimal digit " + hexdec.charAt(i));

            }
        }
        return retStr.toString();
}

public static String group4(String binValue) {
        StringBuilder result = new StringBuilder();
        String tempBin = binValue;

        if (binValue.charAt(0) == '1' && binValue.charAt(1) == '1' && binValue.charAt(2) == '1'
                && binValue.charAt(3) == '1') {
            result.append("B");
            tempBin = "ZZZZ" + binValue.substring(4);
            binValue = tempBin;
        }
        if (binValue.charAt(4) == '1' && binValue.charAt(5) == '1' && binValue.charAt(6) == '1'
                && binValue.charAt(7) == '1') {
            result.append("~B");
            tempBin = binValue.substring(0, 4) + "ZZZZ";
            binValue = tempBin;
        }
        if (binValue.charAt(0) == '1' && binValue.charAt(1) == '1' && binValue.charAt(4) == '1'
                && binValue.charAt(5) == '1') {
            result.append("A");
```

```
            binValue = replaceChar(binValue, 0);
            binValue = replaceChar(binValue, 1);
            binValue = replaceChar(binValue, 4);
            binValue = replaceChar(binValue, 5);
        }
        if (binValue.charAt(1) == '1' && binValue.charAt(2) == '1' && binValue.charAt(5) == '1'
                    && binValue.charAt(6) == '1') {
            result.append("C");
            binValue = replaceChar(binValue, 1);
            binValue = replaceChar(binValue, 2);
            binValue = replaceChar(binValue, 5);
            binValue = replaceChar(binValue, 6);
        }
        if (binValue.charAt(2) == '1' && binValue.charAt(3) == '1' && binValue.charAt(6) == '1'
                    && binValue.charAt(7) == '1') {
            result.append("~A");
            binValue = replaceChar(binValue, 2);
            binValue = replaceChar(binValue, 3);
            binValue = replaceChar(binValue, 6);
            binValue = replaceChar(binValue, 7);
        }

        if (binValue.charAt(0) == '1' && binValue.charAt(3) == '1' && binValue.charAt(4) == '1'
                    && binValue.charAt(7) == '1') {
            result.append("~C");
            binValue = replaceChar(binValue, 0);
            binValue = replaceChar(binValue, 3);
            binValue = replaceChar(binValue, 4);
            binValue = replaceChar(binValue, 7);
```

```
		}
		if (result.toString() != "")
			return result.toString() + "|" + binValue;
		return "";
	}

	public static String group2(String binValue) {
		StringBuilder result = new StringBuilder();
		String tempBin = binValue;
		if (binValue.charAt(0) == '1' && binValue.charAt(1) == '1') {
			result.append("AB");
			binValue = replaceChar(binValue, 0);
			binValue = replaceChar(binValue, 1);
		}
		if (binValue.charAt(1) == '1' && binValue.charAt(2) == '1') {
			result.append("+BC");
			binValue = replaceChar(binValue, 1);
			binValue = replaceChar(binValue, 2);
		}
		if (binValue.charAt(2) == '1' && binValue.charAt(3) == '1') {
			result.append("+~AB");
			binValue = replaceChar(binValue, 2);
			binValue = replaceChar(binValue, 3);
		}
		if (binValue.charAt(4) == '1' && binValue.charAt(5) == '1') {
			result.append("+A~B");
			binValue = replaceChar(binValue, 4);
			binValue = replaceChar(binValue, 5);
		}
```

```
if (binValue.charAt(5) == '1' && binValue.charAt(6) == '1') {
        result.append("+~BC");
        binValue = replaceChar(binValue, 5);
        binValue = replaceChar(binValue, 6);
}
if (binValue.charAt(6) == '1' && binValue.charAt(7) == '1') {
        result.append("+~A~B");
        binValue = replaceChar(binValue, 6);
        binValue = replaceChar(binValue, 7);
}
if (binValue.charAt(0) == '1' && binValue.charAt(4) == '1') {
        result.append("+A~C");
        binValue = replaceChar(binValue, 0);
        binValue = replaceChar(binValue, 4);
}

if (binValue.charAt(1) == '1' && binValue.charAt(5) == '1') {
        result.append("+AC");
        binValue = replaceChar(binValue, 1);
        binValue = replaceChar(binValue, 5);
}
if (binValue.charAt(2) == '1' && binValue.charAt(6) == '1') {
        result.append("+~AC");
        binValue = replaceChar(binValue, 2);
        binValue = replaceChar(binValue, 6);
}
if (binValue.charAt(3) == '1' && binValue.charAt(7) == '1') {
        result.append("+~A~C");
        binValue = replaceChar(binValue, 3);
```

```
                binValue = replaceChar(binValue, 7);
        }

        if (binValue.charAt(0) == '1' && binValue.charAt(3) == '1') {
                result.append("+B~C");
                binValue = replaceChar(binValue, 0);
                binValue = replaceChar(binValue, 3);
        }
        if (binValue.charAt(4) == '1' && binValue.charAt(7) == '1') {
                result.append("+~B~C");
                binValue = replaceChar(binValue, 4);
                binValue = replaceChar(binValue, 7);
        }

        if (!(result.toString().equals(""))) {
                String tempRes = result.toString();

                if (tempRes.charAt(0) == '+')
                        tempRes = tempRes.substring(1);

                return tempRes + "|" + binValue;
        }
        return "";
}

public static String group1(String binValue) {
        StringBuilder result = new StringBuilder();
        String tempBin = binValue;
```

```
if (binValue.charAt(0) == '1') {
        result.append("AB~C");
        tempBin = "Z" + binValue.substring(1, 8);
        binValue = tempBin;

}
if (binValue.charAt(1) == '1') {
        result.append("+ABC");
        binValue = replaceChar(binValue, 1);
}
if (binValue.charAt(2) == '1') {
        result.append("+~ABC");
        binValue = replaceChar(binValue, 2);
}
if (binValue.charAt(3) == '1') {
        result.append("+~AB~C");
        binValue = replaceChar(binValue, 3);
}
if (binValue.charAt(4) == '1') {
        result.append("+~A~B~C");
        binValue = replaceChar(binValue, 4);
}
if (binValue.charAt(5) == '1') {
        result.append("+A~BC");
        binValue = replaceChar(binValue, 5);
}
if (binValue.charAt(6) == '1') {
        result.append("+~A~BC");
        binValue = replaceChar(binValue, 6);
```

```
		}
		if (binValue.charAt(7) == '1') {
			result.append("+~A~B~C");
			binValue = replaceChar(binValue, 7);
		}
		if (!(result.toString().equals(""))) {
			String tempRes = result.toString();

			if (tempRes.charAt(0) == '+')
				tempRes = tempRes.substring(1);

			return tempRes + "|" + binValue;
		}
		return "";
	}

	public static String replaceChar(String str, int index) {
		StringBuilder myStr = new StringBuilder(str);
		myStr.setCharAt(index, 'Z');
		return myStr.toString();
	}

}
```