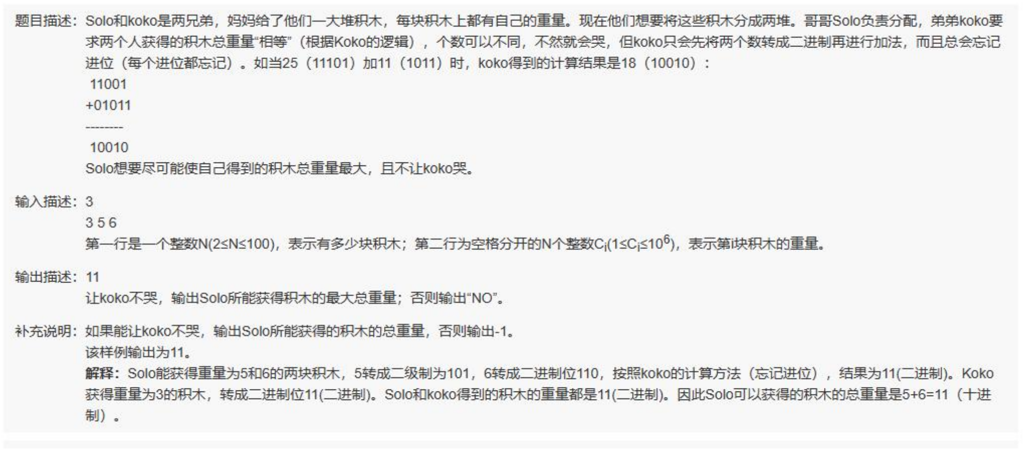
分积木



题目描述：Solo和koko是两兄弟，妈妈给了他们一大堆积木，每块积木上都有自己的重量。现在他们想要将这些积木分成两堆。哥哥Solo负责分配，弟弟koko要求两个人获得的积木总重量“相等”（根据Koko的逻辑），个数可以不同，不然就会哭，但koko只会先将两个数转成二进制再进行加法，而且总会忘记进位（每个进位都忘记）。如当25（11101）加11（1011）时，koko得到的计算结果是18（10010）：

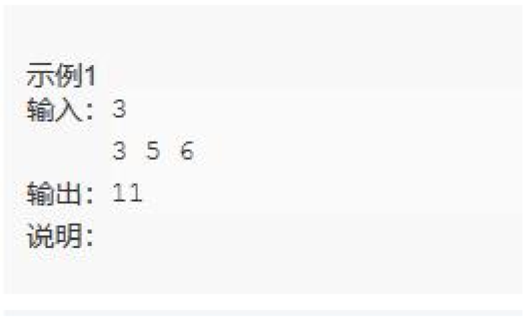
```
 11001
+01011
--------
 10010
```

Solo想要尽可能使自己得到的积木总重量最大，且不让koko哭。

输入描述：3
3 5 6
第一行是一个整数N(2≤N≤100)，表示有多少块积木；第二行为空格分开的N个整数C_i(1≤C_i≤10^6)，表示第i块积木的重量。

输出描述：11
让koko不哭，输出Solo所能获得积木的最大总重量；否则输出“NO”。

补充说明：如果能让koko不哭，输出Solo所能获得的积木的总重量，否则输出-1。
该样例输出为11。
解释：Solo能获得重量为5和6的两块积木，5转成二级制为101，6转成二进制位110，按照koko的计算方法（忘记进位），结果为11(二进制)。Koko获得重量为3的积木，转成二进制位11(二进制)。Solo和koko得到的积木的重量都是11(二进制)。因此Solo可以获得的积木的总重量是5+6=11（十进制）。



示例1
输入：3
　　　3 5 6
输出：11
说明：

```java
import java.util.Scanner;
import java.io.*;

// 注意类名必须为 Main, 不要有任何 package xxx 信息
public class Main {
    public static void main(String[] args) throws IOException {
        BufferedReader bf = new BufferedReader(new InputStreamReader(System.in));
        String str;
        while((str=bf.readLine())!=null){
            int num=Integer.parseInt(str);
            String[] arr=bf.readLine().split(" ");
            int width= arr.length;
            int soloWeight=0;
            int totalFaithWeight=0;
            int minWeight=Integer.MAX_VALUE;
            int[] weights=new int[width];
            for (int i = 0; i < width; i++) {
                weights[i]=Integer.parseInt(arr[i]);
                totalFaithWeight=totalFaithWeight^weights[i];//对所有数求异或运算
                soloWeight+=weights[i];
                minWeight=Math.min(minWeight,weights[i]);
            }
            if (totalFaithWeight==0){
                System.out.println(soloWeight-minWeight);
```

```
                  }else{
                        System.out.println(-1);
                  }
            }
      }
}
```