Unfair Game

by HackerRank

You are playing a game of Nim with a friend. The rules are are follows:

1) Initially, there are N piles of stones. Two players play alternately.

2) In each turn, a player can choose one non empty pile and remove any number of stones from it. At least one stone must be removed.

3) The player who picks the last stone from the last non empty pile wins the game.

It is currently your friend's turn. You suddenly realize that if your friend was to play optimally in that position, you would lose the game. So while he is not looking, you decide to cheat and add some (possibly 0) stones to each pile. You want the resultant position to be such that your friend has no guaranteed winning strategy, even if plays optimally. You cannot create a new pile of stones. *You can only add stones, and not remove stones from a pile*. What is the least number of stones you need to add?

Input Format

The first line contains the number of cases T. T cases follow. Each case contains the number N on the first line followed by N numbers on the second line. The ith number denotes si, the number of stones in the ith pile currently.

Constraints

1 <= T <= 20

2 <= N <= 15

1 <= si < 1000000000 (10^9)

Output Format

Output T lines, containing the answer for each case. If the current position is already losing for your friend, output 0.

Sample Input

3

2

1 3

3

1 1 1

4

10 4 5 1

Sample Output

2

3

6

Explanation

For the first case, add 2 stones to the first pile. Then, both piles will have 3 stones each. It is easy to verify that your friend cannot win the game unless you make a mistake.

For the second case, add 1 stone to the first pile, and 2 stones to the second pile.

Submissions:428

Max Score:100
Difficulty: Advanced

Rate This Challenge:
☆☆☆☆☆

More

Current Buffer (saved locally, editable)

Java 7

```
/* Sample program illustrating input/output */

import java.util.*;

class Solution{
   public static void main( String args[] ){

// helpers for input/output

      Scanner in = new Scanner(System.in);

      int test;
      test = in.nextInt();
      for(int t=0; t<test; t++){
         int N;
         N = in.nextInt();

         int V[] = new int[N];
         for(int i=0; i<N; i++){
            V[i] = in.nextInt();
         }

         int result = 0;
         System.out.println( result );
      }
   }
}

```

Line: 1 Col: 1

Upload Code as File

Test against custom input

Run Code Submit Code