Algorithm Design Laboratory with Applications

Prof. Stefano Leucci

Problem: *Candies.*

Alice bought $n \geq 2$ candy packets for her two children Bob and Charlie. The packets are indexed with the integers from 1 to n and the i-th packet contains $c_i \in \mathbb{N}^+$ candies. Bob and Charlie are stubborn kids: neither of them wants to share candies from the same packet with his brother. After hours of arguing Bob and Charlie still haven't agreed on how to share the packets, so Alice decides to settle the dispute using the power of algorithms!

Help Alice design an algorithm that, given n and the integers $c_1, \dots, c_n$, finds a way to assign each packet to either Bob or Charlie so that the number of candies received by each kid is as close as possible. Formally, Alice wants to partition the set of indices $\{1, \dots, n\}$ into two sets B and C so that $\Delta(B, C) = \left|\sum_{i \in B} c_i - \sum_{i \in C} c_i\right|$ is minimized (a perfectly fair division corresponds to the case $\Delta(B, C) = 0$).

Input. The input consists of a set of instances, or *test-cases*, of the previous problem. The first line contains the number T of test-cases. Each test-case consists of two lines. The first line contains the integer n, while the second line contains the n integers $c_1, \dots, c_n$.

Output. The output consists of T lines. The i-th line is the answer to the i-th test-case and contains the minimum attainable value of $\Delta(B, C)$.

Assumptions. $1 \leq T \leq 10$; $1 \leq n \leq 40$; $\forall i = 1, \dots, n$, $1 \leq c_i \leq 2^{40}$.

Example.

Input (corresponding to the above picture):

```
2
6
2 12 4 10 8 4
4
4 8 1 7
```

Output:

```
0
2
```

Requirements. Your algorithm should require $O^*(2^{\frac{n}{2}})$ time (with reasonable hidden polynomial factors in n). The algorithm's time complexity **should not** be a function of the overall number $\sum_{i=1}^{n} c_i$ of candies.

Notes. A reasonable implementation should not require more than 3 seconds for each input file.