

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 $N(2 \leq N \leq 100)$ ，表示有多少块积木；第二行为空格分开的 N 个整数 $C_i(1 \leq C_i \leq 10^6)$ ，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* 转成二进制为 *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：

```
#include <iostream>
```

```
#include <cstdio>
```

```
#include <vector>
```

```
using namespace std;
```

```
int main() {
```

```
    int n;
```

```
    int tmp;
```

```
    int xor_sum;
```

```
    vector<int> arr;
```

```
    scanf("%d\n", &n);
```

```
    for (int i = 0; i < n ; i++) {
```

```
        cin >> tmp;
```

```
        if (i == 0) {
```

```
        xor_sum = tmp;

    } else {

        xor_sum ^= tmp;

    }

    arr.push_back(tmp);
}

if (xor_sum != 0) {

    cout << "-1" << endl;

    return 0;

}

int sum = 0;

int min_elem = arr[0];

for (int i = 0 ; i < arr.size(); i++) {

    sum += arr[i];

    min_elem = min(min_elem, arr[i]);

}

cout << sum - min_elem << endl;

}

// 64 位输出请用 printf("%lld")
```