A2. Educational Game

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The Smart Beaver from ABBYY began to develop a new educational game for children. The rules of the game are fairly simple and are described below.

The playing field is a sequence of n non-negative integers a_i numbered from 1 to n. The goal of the game is to make numbers $a_1, a_2, ..., a_k$ (i.e. some prefix of the sequence) equal to zero for some fixed k $(k < n)$, and this should be done in the smallest possible number of moves.

One move is choosing an integer i $(1 \le i \le n)$ such that $a_i > 0$ and an integer t $(t \ge 0)$ such that $i + 2^t \le n$. After the values of i and t have been selected, the value of a_i is decreased by 1, and the value of a_{i+2^t} is increased by 1. For example, let $n = 4$ and $a = (1, 0, 1, 2)$, then it is possible to make move $i = 3$, $t = 0$ and get $a = (1, 0, 0, 3)$ or to make move $i = 1$, $t = 1$ and get $a = (0, 0, 2, 2)$ (the only possible other move is $i = 1$, $t = 0$).

You are given n and the initial sequence a_i. The task is to calculate the minimum number of moves needed to make the first k elements of the original sequence equal to zero for each possible k $(1 \le k < n)$.

Input

The first input line contains a single integer n. The second line contains n integers a_i $(0 \le a_i \le 10^4)$, separated by single spaces.

The input limitations for getting 20 points are:

- $1 \le n \le 300$

The input limitations for getting 50 points are:

- $1 \le n \le 2000$

The input limitations for getting 100 points are:

- $1 \le n \le 10^5$

Output

Print exactly n - 1 lines: the k-th output line must contain the minimum number of moves needed to make the first k elements of the original sequence a_i equal to zero.

Please do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams, or the `%I64d` specifier.

Examples

input

```
4
1 0 1 2
```

output

```
1
1
3
```

input

```
8
1 2 3 4 5 6 7 8
```

output

```
1
3
6
10
16
24
40
```