Ambitious Manager

The Bogus Corporation distributes salary to its employees in a weird manner. The salary is distributed every **K** days, and instead of same salary for each day, the salary for the i^{th} day is a_i. An ambitious young manager, fresh from *Institute of Mismanagement*, observes that people usually prefer to take leave towards the end of this period of K days, when the workload is higher. Instead of revising each of the a_i's, the manager comes up with a quick fix solution - he redefines the new salary on the i^{th} day as $b_i = a_i + 2a_{i-1} + 2^2a_{i-2} + 2^3a_{i-3} + \ldots + 2^{i-1}a_1$. Baba, one of the employees, is in a dire financial crisis, and must accumulate at least **N** rupees at the end of the forthcoming period. Being a lazy worker that he is, he is interested in finding out if attending particular days would guarantee him *exactly* **N** rupees at the end of the period. Can you help Baba?

Input

First line contains a single integer integer **T**, the number of test cases (1 <= T <= 100). Each test case is described on two lines. First line contains two integers, **N** and **K** (1 <= N <= 2^{63}-1, 1 <= K <= 50) , the second line contains a space separated list of K integers, the $\mathbf{a_i}$'s (1 <= a_i <= 1000).

Output

For each test case, output on a single line 1-based indices of the days (separated by a single space) he should attend to ensure a salary of exactly N rupees at the end of the period. The indices should be printed in the sorted order. In case of multiple answers, output any one of them. If there is no answer, print -1.

Example

Input:

```
2
9 3
1 1 2
10 2
2 3
```

Output:

```
1 3
-1
```