G. Good Key, Bad Key

time limit per test: 3 seconds
memory limit per test: 256 megabytes

There are n chests. The i-th chest contains a_i coins. You need to open all n chests **in order from chest 1 to chest n**.

There are two types of keys you can use to open a chest:

- a good key, which costs k coins to use;
- a bad key, which does not cost any coins, but will halve all the coins in each unopened chest, **including the chest it is about to open**. The halving operation **will round down** to the nearest integer for each chest halved. In other words using a bad key to open chest i will do $a_i = \lfloor \frac{a_i}{2} \rfloor, a_{i+1} = \lfloor \frac{a_{i+1}}{2} \rfloor, \dots, a_n = \lfloor \frac{a_n}{2} \rfloor$;
- any key (both good and bad) breaks after a usage, that is, it is a one-time use.

You need to use in total n keys, one for each chest. Initially, you have no coins and no keys. If you want to use a good key, then you need to buy it.

During the process, you are allowed to go into debt; for example, if you have 1 coin, you are allowed to buy a good key worth $k = 3$ coins, and your balance will become -2 coins.

Find the maximum number of coins you can have after opening all n chests in order from chest 1 to chest n.

Input

The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of test cases.

The first line of each test case contains two integers n and k ($1 \leq n \leq 10^5; 0 \leq k \leq 10^9$) — the number of chests and the cost of a good key respectively.

The second line of each test case contains n integers a_i ($0 \leq a_i \leq 10^9$) — the amount of coins in each chest.

The sum of n over all test cases does not exceed 10^5.

Output

For each test case output a single integer — the maximum number of coins you can obtain after opening the chests in order from chest 1 to chest n.

Please note, that the answer for some test cases won't fit into 32-bit integer type, so you should use at least 64-bit integer type in your programming language (like `long long` for C++).

Example

input

```
5
4 5
10 10 3 1
1 2
1
3 12
10 10 29
12 51
5 74 89 45 18 69 67 67 11 96 23 59
2 57
85 60
```

output

```
11
0
13
60
58
```

Note

In the first test case, one possible strategy is as follows:

- Buy a good key for 5 coins, and open chest 1, receiving 10 coins. Your current balance is $0 + 10 - 5 = 5$ coins.
- Buy a good key for 5 coins, and open chest 2, receiving 10 coins. Your current balance is $5 + 10 - 5 = 10$ coins.
- Use a bad key and open chest 3. As a result of using a bad key, the number of coins in chest 3 becomes $\left\lfloor \frac{3}{2} \right\rfloor = 1$, and the number of coins in chest 4 becomes $\left\lfloor \frac{1}{2} \right\rfloor = 0$. Your current balance is $10 + 1 = 11$.
- Use a bad key and open chest 4. As a result of using a bad key, the number of coins in chest 4 becomes $\left\lfloor \frac{0}{2} \right\rfloor = 0$. Your current balance is $11 + 0 = 11$.

At the end of the process, you have 11 coins, which can be proven to be maximal.