

# Zipf's Song

Problem ID: zipfsong

Your slightly pointy-bearded boss has assigned you to write software to find the best songs from different music albums. And the software should be finished in an hour. But don't panic, you don't have to solve the problem of writing an AI with good taste. At your disposal is the impeccable taste of a vast horde of long-tailed monkeys. Well, at least almost. The monkeys are not very communicative (or rather, you're not sure which song "Ook!" is supposed to refer to) so you can't ask them which songs are the best. What you can do however is to look at which songs the monkeys have listened to and use this information to deduce which songs are the best.

At first, you figure that the most listened to songs must be the best songs. However, you quickly realize that this approach is flawed. Even if all songs of the album are equally good, the early songs are more likely to be listened to more often than the later ones, because monkeys will tend to start listening to the first song, listen for a few songs and then, when their fickle ears start craving something else, stop listening. Instead, if all songs are equal, you expect that their play frequencies should follow *Zipf's Law*.

Zipf's Law is an empirical law originally formulated about word frequencies in natural languages, but it has been observed that many natural phenomena, such as population sizes and incomes, approximately follow the same law. It predicts that the relative frequency of the  $i$ 'th most common object (in this case, a song) should be proportional to  $1/i$ .

To illustrate this in our setting, suppose we have an album where all songs are equally good. Then by Zipf's Law, you expect that the first song is listened to twice as often as the second song, and more generally that the first song is listened to  $i$  times as often as the  $i$ 'th song. When some songs are better than others, those will be listened to more often than predicted by Zipf's Law, and those are the songs your program should select as the good songs. Specifically, suppose that song  $i$  has been played  $f_i$  times but that Zipf's Law predicts that it would have been played  $z_i$  times. Then you define the quality of song  $i$  to be  $q_i = f_i / z_i$ . Your software should select the songs with the highest values of  $q_i$ .

Input

The first line of input contains two integers  $n$  and  $m$  ( $1 \leq n \leq 50000$ ,  $1 \leq m \leq n$ ), the number of songs on the album, and the number of songs to select. Then follow  $n$  lines. The  $i$ 'th of these lines contains an integer  $f_i$  and string  $s_i$ , where  $0 \leq f_i \leq 10^{12}$  is the number of times the  $i$ 'th song was listened to, and  $s_i$  is the name of the song. Each song name is at most 30 characters long and consists only of the characters 'a'-'z', '0'-'9', and underscore ('\_').

## Output

Output a list of the  $m$  songs with the highest quality  $q_i$ , in decreasing order of quality. If two songs have the same quality, give precedence to the one appearing first on the album (presumably there was a reason for the producers to put that song before the other).

### Sample input 1

Play Count	Song
30	One
30	Two
15	Three
25	Four

### Sample output 1

four

two

### Sample input 2

15	3
197812	5 4
189517	re_hash
78906	5 4
189518	tomorrow_comes_today
39453	new_genious
210492	clint_eastwood
26302	man_research
22544	punk
19727	sound_check
17535	double_bass
18782	rock_the_house
198189	19_2000
13151	latin_simone
12139	starshine
11272	slow_country
10521	m1_al

### Sample output 2

19\_2000

clint\_eastwood

tomorrow\_comes\_today