Fibonacci Colonies (fibonaccibug)

Bug colonies have been the center of attention of scientists for a long time. Through some technological advancements, we are now able to describe a bug colony using a number known as the *degree* of the colony. A colony of degree 0 or 1 represents a colony with one bug. A colony of degree $i > 1$ is obtained by merging a colony of degree $i - 1$ together with a colony of degree $i - 2$. As such, a colony of degree 2 has two bugs, a colony of degree 3 has three bugs, a colony of degree 4 has five bugs and so on.

Marco owns the biggest bug farm in the world, having at his disposal a virtually infinite amount of colonies of any degree. Every day he receives N offers, each described by two numbers A_i and B_i, meaning that he can sell as many colonies of degree A_i as he wants and get B_i money for each colony of that degree. Unfortunately, the antitrust laws on the bug trading market forbid him to sell more than K bugs in a single day overall (selling a colony is equivalent to selling all the bugs in that colony). Given the description of T days, if he optimally chooses which offers to accept, what is the maximum amount of money Marco can obtain in each day?

☞ Among the attachments of this task you may find a template file `fibonaccibug.*` with a sample incomplete implementation.

Input

The first line contains one integer T, the number of days. The following lines contain the description of each day. For each day, the first line contains two integers N and K, the number of offers and the maximum number of bugs you can sell that day. The following N lines contain contain two integers A_i and B_i, the colony of the offer and the price per colony.

Output

You need to write T lines, each with an integer: the maximum profit you can make for each day.

Constraints

- $1 \leq T,\ N,\ K \leq 100\,000$.
- $0 \leq A_i \leq 100\,000$.
- $1 \leq B_i \leq 10^9$.
- The sum of all N and all K across the days of a single input does not exceed $201\,000$.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points) Examples.

– **Subtask 2** (10 points) $T = 1,\ N \leq 6,\ K \leq 6,\ A_i \leq 10$.

– **Subtask 3** (10 points) $K = 1$.

– **Subtask 4** (35 points) $N, K \leq 5500$ and $A_i \leq 11\,000$.

– **Subtask 5** (45 points) No additional limitations.

Examples

input	output

1
5 11
1 2
2 2
3 5
4 9
5 50
	56

2
3 10
1 10
4 60
3 40
2 10
1 30
2 40
	130
	300

Explanation

In the **first sample case** it is optimal to choose the fifth offer once and the first one three times.

In the **second sample case**, for the first day it is optimal to choose the first offer once and the third offer three times; for the second day it is optimal to choose ten times the first offer.