Knapsack Auction: VCG Solution

n buyers want to compete for m items through auction.

The i-th buyer want to buy c_i items at a price of v_i. Note that v_i also means the i-th buyer's **valuation** of the items.

The VCG auction allocates the m items which maximize the sum of the valuations of those who gets c_i items as he/she wants.

- $V = (v_1, v_2, \ldots, v_n)$;
- $C = (c_1, c_2, \ldots, c_n)$;
- $\mathrm{VA}(V, C) = \{A \in 2^{\{1,2,\ldots,n\}} : \sum_{i \in A} c_i \leqslant m\}$, which means the set of all **Valid Allocations** to the n buyers. Valid means that m items are enough to be allocated.
- $\mathrm{MSV}(V, C) = \max_{A \in \mathrm{VA}(V,C)} \sum_{i \in A} v_i$, which means the **Maximal Sum of the Valuations** of those who gets c_i items as he/she wants.
- $A_M(V, C) \in \arg\max_{A \in \mathrm{VA}(V,C)} \sum_{i \in A} v_i$, which means the valid allocation that maximize the sum of the valuations. The buyers in $A_M(V, C)$ win the auction and get the items as they want.
- $V_{-i} = (v_1, v_2, \ldots, v_{i-1}, v_{i+1}, \ldots, v_n)$, which means excluding v_i from V;
- $C_{-i} = (c_1, c_2, \ldots, c_{i-1}, c_{i+1}, \ldots, c_n)$, which means excluding c_i from C;
- $p_i = v_i + \mathrm{MSV}(V_{-i}, C_{-i}) - \mathrm{MSV}(V, C)$, which is the definition of the **payment** of the i-th buyer $i \in A_M(V, C)$ if he/she gets the c_i items.

Your task is to compute $A_M(V, C)$ and their payments. If there are more than one choice of $A_M(V, C)$, please choose the lexicographically least one.

Input format

On the first line, two positive integers n and m, separated by a space.

For the following n lines:

- On the i-th line, two positive integers v_i and c_i, separated by a space.

Output format

On the first line, print $A_M(V, C)$ (lexicographically least one). The numbers are separated by spaces.

On the second line, print p_i for all $i \in A_M(V, C)$. The numbers are separated by spaces.

Testing on OJ

There are 20 testcases on OJ. For the first 10 testcases, your program (assuming it is named `foo.cc`) is compiled with

```
g++ foo.cc -o foo -std=c++20 -Wall -Wpedantic -Wextra -Werror -DONLINE_JUDGE -
fmax-errors=3 -fdiagnostics-color=always -fsanitize=undefined -fsanitize=address
-g
```

For testcases 11 ~ 20, your program is compiled with

```
g++ foo.cc -o foo -std=c++20 -Wall -Wpedantic -Wextra -Werror -DONLINE_JUDGE -fmax-errors=3 -fdiagnostics-color=always -O2
```

Due to the OJ settings, there is no way of running a single testcase in pretest (运行自测). You can enter your input data there and see the output. For pretesting, the program will be compiled with the same command as for the first 10 testcases.

Data constraints

- $1 \leqslant n, m \leqslant 5000$.
- $1 \leqslant v_i \leqslant 10^9, 1 \leqslant c_i \leqslant m$

Examples

Input 1

```
2 2
2 1
3 2
```

Output 1

```
2
2
```

Explanation 1

- $v_1 = 2, c_1 = 1$;
- $v_2 = 3, c_2 = 2$;
- $\mathrm{MSV}(V, C) = 3, A_M(V, C) = \{2\}$;
- $\mathrm{MSV}(V_{-2}, C_{-2}) = 2, A_M(V_{-2}, C_{-2}) = \{1\}, p_2 = 3 + 2 - 3 = 2$.

Input 2

```
4 4
1 1
2 2
3 3
2 2
```

Output 2

```
1 3
1 3
```

Explanation 2

- $v_1 = 1, c_1 = 1$;
- $v_2 = 2, c_2 = 2$;
- $v_3 = 3, c_1 = 3$;
- $v_4 = 2, c_2 = 2$;

- $\mathrm{MSV}(V, C) = 4, A_M(V, C) \in \{\{1, 3\}, \{2, 4\}\}$, choose the lexicographically least one $\{1, 3\}$
- $\mathrm{MSV}(V_{-1}, C_{-1}) = 4, A_M(V_{-1}, C_{-1}) = \{2, 4\}, p_1 = 1 + 4 - 4 = 1$;
- $\mathrm{MSV}(V_{-3}, C_{-3}) = 4, A_M(V_{-3}, C_{-3}) = \{2, 4\}, p_3 = 3 + 4 - 4 = 3$.

提交与评分

本题的评分由 OJ 分数（60%）和线下 check （40%）两部分构成。线下 check 会在此次作业结束时间之后进行。

注：线下 check 也带有检查学术诚信的含义，当然这不是唯一的手段。如果被认定为抄袭， OJ 的分数也会作废，并且会有惩罚。**特别强调，抄袭来自 generative AI 的代码和抄袭网上的代码是同等处理的，我们建议您在写作业时关闭一切generative AI 工具。**