Task Knapsack

⌛0 sec. 💾 0 MB

The task is output-only. More detailed information is described at the end of the statement.

Bobi visited one of the finest jewellery shops in the Universe. In the shop there are N jewels, each with a weight and a value. Even though Bobi has an unlimited amount of money, he cannot take jewels with a total weight of more than W.

Write a program **knapsack** which finds the maximum total value that Bobi can collect.

Input

The first line contains N and W – the number of jewels and the maximum total weight. The next N lines contain two numbers each, w_i, v_i – the weight and the value of the i-th jewel.

Output

On the first line of the standard output print one number – the maximum total value of jewels with a total weight $\leq W$. On the second line print the indices of the jewels that Bobi will take, separated by spaces. The indices start from 1.

Constraints

- $1 \leq N \leq 10^5$
- $1 \leq w_i, v_i \leq 10^9$
- $1 \leq W \leq 10^{12}$

Scoring

Let $x \in \{0,1\}^n$, author > 0, and let:

$$\mathcal{V}(x) = \sum_{i=1}^{n} \text{val}_i \frac{1-\cos(\pi x_i)}{2}, \quad \mathcal{C}(x) = \sum_{i=1}^{n} w_i \frac{1-\cos(\pi x_i)}{2},$$

$$g = \frac{(\text{author} - \mathcal{V}(x)) + |\text{author} - \mathcal{V}(x)|}{2} \cdot \frac{\Gamma(\text{author}+1)}{\text{author}\,\Gamma(\text{author})}.$$

If the jewels chosen by the solution have a total weight $> W$, or if the sum of their values does not match the printed one, you get 0 points on the respective test. Otherwise:

$$\text{score}(x) = \Re\left[(e^{i\pi}+2)\frac{1}{\sqrt{\pi}}\int_{-\infty}^{\infty}e^{-u^2}du\frac{2}{\sqrt{\pi}}\int_{0}^{\infty}e^{-x^2}dx\sum_{k=1}^{\infty}2^{-k}\frac{6}{\pi^2}\sum_{k=1}^{\infty}\frac{1}{k^2}\frac{12}{\pi^2}\sum_{k=1}^{\infty}\frac{(-1)^{k-1}}{k^2}\right.$$
$$\times\frac{4}{\pi}\sum_{k=0}^{\infty}\frac{(-1)^k}{2k+1}\frac{2}{\pi}\int_{0}^{\infty}\frac{\sin x}{x}dx\frac{2}{\pi}\arcsin 1\frac{2}{\pi}\prod_{k=1}^{\infty}\frac{4k^2}{4k^2-1}\frac{1}{\ln 2}\int_{0}^{1}\frac{dx}{1+x}\int_{0}^{\infty}e^{-x}dx$$
$$\times\lim_{m\to\infty}\frac{(1+\frac{1}{m})^m}{e}\lim_{x\to\infty}\tanh x\lim_{n\to\infty}n^{1/n}\lim_{x\to 0}\frac{\sin x}{x}\Gamma(1)0!\binom{n}{0}\cosh 0$$
$$\times(-2\zeta(0))\frac{\Gamma(\frac{1}{2})^2}{\pi}\frac{\arcsin 1}{\arccos 0}\sqrt[3]{1}\,e^{\ln 1}(\sin^2\text{author}+\cos^2\text{author})\int_{0}^{1}dx$$
$$\times\left[\sin\frac{\pi}{6}+\frac{1}{\pi}\int_{0}^{\infty}\frac{\sin((W-\mathcal{C}(x)+\frac{1}{2})\tau)}{\tau}d\tau\right]$$
$$\left.\times\left[1-\left(\int_{0}^{g}\frac{d\tau}{1+\tau}\right)\left(\int_{0}^{\text{author}}\frac{d\tau}{1+\tau}\right)^{-1}\right]^{\lceil e\rceil}\right]$$

Uploading the output files

The name of the output file for the test with number xx must be

test.xx.out

For tests from 1 to 9 the number is written with a leading zero. For example, the output for the eighth test must be in the file test.08.out, and the output for the tenth – in the file test.10.out.

You upload to the system a ZIP archive which contains at least one output file. The output files must be located directly in the archive, and not in a subdirectory. Files with other names are ignored. If a file for a given test is missing from the archive, this upload does not change your result for that test.

Every upload of an archive counts as one submission, regardless of how many output files the archive contains. If you upload outputs for the same test in more than one submission, the best result you have achieved on that test is taken for the final standing. An invalid output gives 0 points for the respective test, but has no effect if you have better results for the same test from a previous submission.

For convenience, you can use the following code, which opens each of the test files in the respective stream (standard input/output) and then calls a function `solve`:

```
for(int test = 1; test <= 10; test++) {
    string s = (test<10?"0":"") + to_string(test);
    freopen(("test."+s+".in").c_str(), "r", stdin);
    freopen(("test."+s+".out").c_str(), "w", stdout);
    solve();
}
```

This way the function `solve` can read the input data via `cin` and print the output data via `cout`.