INSTITUTO SUPERIOR TÉCNICO

Analysis and Synthesis of Algorithms 2023/2024

1° Project

Date announced: November 14, 2023
Deadline for submission: December 04, 2023

Problem description

Engineer João Caracol was hired by the SuperMarble factory to optimize one of its marble slab cutting lines. The line receives a slab of marble that has to be cut in order to produce pieces with the dimensions required by the factory's customers. The line has a two-disc machine that can cut slabs from one side to the other. The cutting process works as follows:

- the sheet is cut vertically or horizontally;
- each of the two new sheets produced re-enters the cutting line or leaves the line if it matches the dimensions of one of the parts to be produced or can no longer be converted into a part.

The factory is currently able to dispose of all production, so priority should be given to manufacturing higher-value parts.

The aim of Eng. Caracol is to build a program that, given a marble slab, calculates the maximum value that can be obtained from it by cutting it into pieces corresponding to the dimensions requested by customers. Eng. Caracol can produce several copies of the same piece as it sees fit. More specifically: the line receives a rectangular slab of marble with dimensions $X \times Y$. In addition, the Snail Engineer has access to a list of the n types of parts to be produced, all with different dimensions. Each type of piece $i \in \{1,..., n\}$ corresponds to a rectangle of marble with dimensions $a_i \times b_i$ and is sold at a price p $._i$

Input

The input file contains the dimensions of the sheet to be cut and the dimensions of the various types of parts requested; these values are represented as follows:

- the first line contains two positive integers X and Y, separated by a blank space, which correspond to the dimensions of the plate;
- the second line contains a positive integer n, which corresponds to the number of types of parts that can be produced;
- n lines describing each of the i types of parts that can be produced; each line is made up of three positive integers a_i , b_i and p_i separated by a blank space, where $a_i \times b_i$ correspond to the dimensions of the type of part and p_i to the price of the part.

Output

You should write in the output the maximum value that can be obtained from the sheet given as input; if no part can be produced, you should simply print 0.

Examples

Input

```
1 3
2
1 1 1
1 3 10
```

Output 1

```
10
```

Input 2

```
3 20
3
2 2 4
1 5 10
3 7 20
```

Output 2

```
120
```

1Input 3

```
7 4
2
6 3 130
1 2 5
```

Output 3

```
155
```

Input 4

```
4 3
2
3 3 10
3 2 6
```

Output 4

```
12
```

Implementation

The project should preferably be implemented using the C, C++ or Rust programming languages. Submissions in interpreted languages (Java/Python) will also be accepted, although strongly discouraged. Students who choose to do so should be aware that submissions in Java/Python may not pass all the tests even if they implement the correct algorithm.
The time needed to implement this project is less than 15 hours.

Compilation parameters:

```
C++: g++ -std=c++11 -O3 -Wall file.cpp -lm
C: gcc -O3 -ansi -Wall file.c -lm
Rust: rustc --edition=2021 -C opt-level=3 file.rs
Javac: javac File.java
Java: java -Xss32m -Xmx256m -classpath . File
Python: python3 file.py
```

Project submission

The project submission should include a summary report and a file with the solution's source code. Information on possible programming languages is available on the Mooshak system website. The programming language is identified by the file extension. For example, a project written in C should have the extension .c. After compilation, **the resulting program should read from the standard input and write to the standard output**. Information on compilation options and restrictions can be obtained from the Mooshak system's help button. The compilation command should not produce output, otherwise it will be considered a compilation error.

Report: must be submitted via the Fénix system in PDF format with no more than 2 pages, 12pt font and 3cm margins. Reports that are not submitted in PDF format will score 0. The report should include a description of the solution, the theoretical analysis and the experimental evaluation of the results. The report must include any references that were used in carrying out the project. An example report template will be released in due course, containing further instructions on its content.

Source code: must be submitted via the Mooshak system and the report (in PDF format) must be submitted via Fénix. The source code will be evaluated automatically by the Mooshak system (`http://acp.tecnico.ulisboa.pt/~mooshak/`). Students are encouraged to submit preliminary solutions to the Mooshak system and Fénix as soon as possible. Please note that only the last submission will be considered for evaluation. All previous submissions will be ignored: this includes the source code and the report.

Evaluation

The project must be carried out in groups of one or two students and will be assessed in two phases. In the first phase, during submission, each implementation will be run through a set of tests, which account for 85% of the final grade. In the second phase, the report will be assessed. The grade for the report contributes 15% of the final grade.

Automatic evaluation

The first phase of the project is evaluated automatically with a set of tests, which are run on a computer with the **GNU/Linux** operating system. It is essential that the source code compiles without errors and respects the input and output standards indicated above. Projects that do not respect the specified formats will be penalized and may score 0 if they fail all the tests. The tests will **not be made public before submission**. However, all tests will be made available after the deadline for submitting the project. In addition to checking the correctness of the output produced, the assessment environment **restricts the** available **memory and execution time.** Most tests run the `diff` command as follows:

```
diff expectedoutput result
```

The `result` file contains the output generated by the executable from the `input` file. The `expectedoutput file` contains the expected output. A program passes a test and receives the corresponding value when the `diff` command reports no differences (i.e. produces no output).
The grade obtained in the automatic classification may be cut if the analysis of the code shows the use of solutions adjusted to specific inputs or random/constant outputs.

Copy Detection

The evaluation of projects includes a procedure for detecting copies. Submitting a project implies a commitment that the work has been carried out exclusively by the students. Violating this commitment or attempting to submit code that was not developed by the group will result in all the students involved (including the students who provided the code) failing the course. Any attempt at cheating, direct or indirect, will be reported to the IST Pedagogical Council, to the course coordinator, and will be penalized in accordance with the rules approved by the University and published in the "Diário da República".