C++ OOP Retake Exam – 27 April 2024

Submit your solutions here: https://judge.softuni.org/Contests/Practice/Index/4741

2. Materials

You're given a list of building materials, which have different properties and types. Each building material may require another building material to be already in the storage, so that it can be produced.

When a new material is read, it is either put in the storage (if it does not require any other materials), or the system tries immediately to produce it, by looking with the needed other materials for completion.

- If there're such materials, they're removed from the storage, and the **newly created material is put in it, as the result of its creation.**
- If there're no such materials, an error message is produced **"Cannot produce: {material type}"**. In this case, no materials are removed from the storage .

Study the provided skeleton and complete the program, so that it executes and produces the desired outputs.

The ideal solution should mandatorily contain a "**Solution.h**" file and if you deem necessary: **a .cpp file as well**.

Input

The input is lines of materials, which are either to be added to the storage, or to be processed.

The input ends with **"end"**:

```
<material type>
<material type> <list of one or more material types, which are necessary for this one to be processed>
end
```

Output

Once the input ends, the system must print out the storage contents, as in the examples below.

If the storage is empty, it should print `Storage: "Empty"`

Example 1

Input	Explanation
Sand Sand	The first two lines add two units of Sand in the storage.
Bricks Sand Sand	Here we produce Bricks by utilizing the two units of Sand. The two units of Sand are removed from the storage and one unit of Bricks is inserted.
Cement Cement Water Water Water Concrete Cement Water Water	Here we first add two Cement and three Water units to the storage. Then we produce one unit of Concrete from one unit of Cement and two units of Water, and add it to the storage. The storage now has one 1xBricks, 1xConcrete, 1xCement and 1xWater.

`Wall Concrete Bricks Cement`	Here we produce one unit of Wall by using one unit of Concrete, Bricks and Cement.
`end`	The program ends, printint out the storage: one unit of Water and one unit of Wall, all in alphabetical order of their type: `Storage: Wall: 1, Water: 1`

Example 2

Input	Explanation

`Wood`
`Wood`
`Iron`
	Adding 2xWood and 1xIron to the storage.

`Nails Iron` `Nails Iron`	Producing 1xNails from 1xIron and adding it to the storage. On the second Nails command we fail to produce the Nails, as there's no Iron in the storage, and the program outputs: `Cannot produce: Nails` The storage now has Nails and two Wood.
`Bracket Nails Wood`	Here we produce one a Bracket from Nails and Wood.
`end`	The program ends, printint out the storage in alphabetical order of their type: `Storage: Bracket: 1, Wood: 1`

Example 3

Input	Explanation
`end`	The storage is empty: `Storage: Empty`