Problem 1 – Collect Resources

Stamat got totally hooked on the newest strategy game and literally can't stop playing it. The game involves building an army and crushing your opponents, but to build an army, you need resources. The game has a rather unusual resource collection system – you are given the available resources and **several** possible ways to collect them. It's up to you to decide what the optimal path is.

The resources are handed to you as an array of elements in format [***resource_quantity resource_quantity*** … ***resource_quantity***]. Valid resources are ***stone, gold, wood*** and ***food.*** When you step on a resource, you collect it. All of them have the same **value,** so **wood_3** is equal to **gold_3. NOTE:** when there is only **one** piece of a given resource, it can be written as **wood** or **wood_1** (both are valid).

You are also given several different ways to collect the resources – **start** and **step.** The **start** is the zero-based position you start from, and **step** is the number of elements you move to the right. If you reach the end of the resource field, you go back to its start. If the element you jump on is a valid resource, you collect it. The process ends when you reach an element you have already collected.

For example, you have resource field **stone_5 gold_3 water_2 wood_7** and **start: 0; step: 2.** You start from **stone_5** and collect 5 resources. You move two elements to the right and step on **water_2**, which is not a valid resource, so you collect nothing. You move another two elements to the right, but this is outside of the element field, so you have to start from the beginning, which is **stone_5** again. It is already **collected**, so you stop with the process. You have gathered 5 resources **total**. **NOTE**: invalid resources are **never collected**.

Write a program that examines several possible ways to collect resources from the same field and outputs the maximum possible collectable quantity.

Input

- On the first line of input, you are given the resource field.
- On the second line, you are given the integer N – the number of collection paths
- On the next N lines you are given the **start** and **step** for the given path – both integers, separated by a space.

Output

- There is one line of output – the maximum possible quantity that can be collected from any of the patterns

Constraints

- The quantity is in range [1 … 100]
- The number of lines N is in range [1 … 10]
- The **start** index is always inside the resource field. The **step** is a valid positive integer.
- Allowed time/memory 0.1s (C#) 0.25s (Java)/16MB

Input	Output

stone_5 gold_2 wood_7 metal_17
2
0 3
0 2
	14
	
	// Comment: 14 is the quantity collected by path 1.
	// Path 2 yields **12** resources