

Technical challenge - Software Engineer

Challenge description

Bloomon has a production facility that produces bouquets. We simplified how the real one - located in Amstelveen - works, for the purpose of this technical challenge:

- It uses *flowers* of different species and sizes as input;
- It produces *bouquets* of different designs and sizes as output;
- The people in the production facility are making those *bouquets* by design specifications that tell how many *flowers* are needed of which kinds;
- The *flowers* arrive into the facility one-by-one, and they can be stored there until there's enough *flowers* to create a *bouquet*.

Your job is to create an application that takes the *bouquet designs* and the stream of *flowers* as an input, and produce the stream of *bouquets* as an output.

The application needs to be a command line application using standard input and output.

It can be written in any of the following languages: Python, Ruby, Go, JavaScript, TypeScript, PHP.

The solution should be submitted in a GitHub / GitLab repository, with full source code and configuration files to build and run it in a Docker container. Please also add "BloomonDev" user to have access to the repo from the beginning.

Completing the challenge should take approximately 4 hours and we expect you to return it in the next couple of days. If you see you're exceeding the 4 hours, you should submit your solution as it is, with a short explanation of what is left and how you would finish the challenge.

Input / output format specifications

- A **flower specie** is identified by a single, lowercase letter: `a - z`;
- A **flower sizes** is indicated by a single, uppercase letter: `L` (large) and `S` (small).
- A **flower** is identified by a **flower specie** and a **flower size**: for example, `rL`.
- A **bouquet name** is indicated by a single, uppercase letter: `A - Z`;
- A **bouquet size** is indicated by a single, uppercase letter: `L` (large) and `S` (small).
- A **bouquet design** is single line of characters with the following format:

```
<bouquet name><bouquet size><flower 1 quantity><flower 1
specie>...<flower N quantity><flower N specie><total quantity of
flowers in the bouquet>
```

Example: `AL8d10r5t30`

- A **bouquet** is single line of characters with the following format:

```
<bouquet name><bouquet size><flower 1 quantity><flower 1
specie>...<flower N quantity><flower N specie>
```

Example: `AL8d10r5t7z`

- The **bouquet design** and **bouquet** formats includes a **bouquet size** but no **flower sizes**. This is because large *bouquets* are only made from large *flowers*, and small *bouquets* are only made from small *flowers*.
- The **flower species** are listed in alphabetic order and only appear once in both **bouquet designs** and **bouquets**.
- The **flower quantities** are always larger than 0 for both **bouquet designs** and **bouquets**.
- The **total quantity of flowers in the bouquet** for **bouquet design** can be bigger than the sum of the **flower quantities**, allowing extra space in the bouquets that can consist of any kind of flowers.
- The **bouquet** does not have **total quantity of flowers in the bouquet** specified, but the sum of the **flower quantities** should be equal to the **total quantity of flowers in the bouquet** of the corresponding **bouquet design**.
- The **input stream** structure will follow this structure:

```
bouquet design1
bouquet design2
<empty line>
flower1
flower2
flower3
...
```

Example:

```
AS3a4b6k20
AL8d10r5t30

aS
aS
bL
rL
tS
...
```

- The **output** should be a **bouquet** every time one can be created from the available **flowers** according to one of the provided **bouquet designs**:

```
bouquet1
bouquet2
...
```

Example:

```
AL8d10r5t7z
AS10a4b6k
...
```

Wrap up

Are you done? Great!! Please let challenge@bloomon.nl know that you are ready, and we will help you with next steps. Thank you for participating in our code challenge!