The purpose of this exercise is to see how you approach a problem, and how you solve it. We're interested to see how you structure your Ruby code, your command of the language and good design and testing principles, bear this in mind throughout.

HINT: Start with a method that accepts a single string argument and returns a string (or a collection) which represents the ordered sequence of jobs (since each job is a single character).

HINT: Brownie points will be given for showing us your working (notes, commit history, some kind of idea how you approached the problem).

HINT: We're pretty keen on tested code.

Have Fun.

The Challenge

Imagine we have a list of jobs, each represented by a character. Because certain jobs must be done before others, a job may have a dependency on another job. For example, a may depend on b, meaning the final sequence of jobs should place b before a. If a has no dependency, the position of a in the final sequence does not matter.

- Given you're passed an empty string (no jobs), the result should be an empty sequence.
- Given the following job structure:

```
a =>
```

The result should be a sequence consisting of a single job a.

- Given the following job structure:

```
a =>
b =>
c =>
```

The result should be a sequence containing all three jobs abc in no significant order.

- Given the following job structure:

```
a =>
b => c
c =>
```

The result should be a sequence that positions c before b, containing all three jobs abc.

- Given the following job structure:

```
a =>
b => c
c => f
d => a
e => b
f =>
```

The result should be a sequence that positions f before c, c before b, b before e and a before d containing all six jobs abcdef.

- Given the following job structure:

```
 a =>
 b =>
 c => c
```

The result should be an error stating that jobs can’t depend on themselves.

- Given the following job structure:

```
a =>
b => c
c => f
d => a
e =>
f => b
```

The result should be an error stating that jobs can’t have circular dependencies.