Assignment 1: Planes

During these troubled times the Groningen airport is mainly used for practice flights and cargo, but has almost no passengers. Some airlines also use the small airport as a cheap parking spot and to get some repairs done.

In this assignment it is your job to determine in which order the planes landing at the airport will be given clearance to leave again. Be careful and do not cause any accidents!

Problem Setup

You can see a simplified map of the airport in Figure 1.1. Planes arrive at and leave from the airport according to the following operation flow:

1. An incoming plane lands on the **main runway** from right to left.
2. The landed plane reaches the **checkup** point where it is inspected to see whether it requires any repairs.
3. If the plane does not need any repairs, then it moves onto the **waiting runway**.
4. If a plane does need repairs, then it moves into the **hangar**.
5. The waiting runway can hold up to seven planes. Whenever the *waiting runway is full*, all planes on the waiting runway are allowed to move to the main runway and depart.
6. The hangar can hold up to five planes. Whenever the *hangar is full*, first the waiting runway is cleared by letting any planes there depart, then all planes in the hangar move onto the waiting runway.
7. At the end of the day, after all planes have landed, all remaining planes leave the airport. First the waiting runway is cleared, then all planes in the hangar move to the waiting runway and then again the waiting runway is cleared.

Your Task

Fortunately, the control tower already makes sure that only one plane will be on the main runway at any time. But it is your job to tell the control tower in which order the planes should be allowed to depart again. Make sure you follow the rules above and the following additional assumptions.

- All planes have a unique name.
- Only one plane lands at a time, hence it never occurs that both the hangar and the waiting runway are full at the same moment.
- The hangar has only one narrow entrance which is also its exit, and planes cannot move past each other within the hangar. For example, to ensure other planes can still enter, the first plane to arrive in the hangar will move as far into the hangar and away from the entrance as possible.
- Also on the waiting runway the planes are not allowed to overtake each other.

Input

The input consists of an unknown number of lines, ended by an empty line. Every line before that contains the unique name of a plane to land, followed by a tab (`'\t'`) and a string “yes” or “no” saying whether the plane needs repairs.

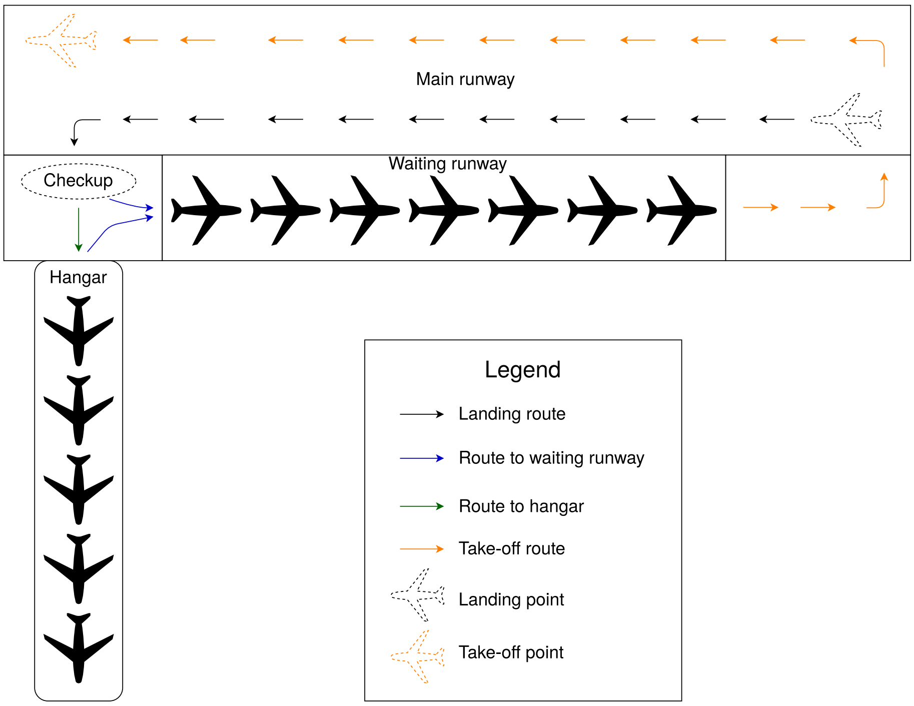

Figure 1.1: Diagram of the airport with indicated routes.

Output

The output gives the clearance order of all the planes which have landed at the airbase. The clearance order consists simply of the original names of n planes, but printed in their corresponding clearance order and separated by a newline.

Hints

- Consider the linear data structures from Lecture 1 and Chapter 2 of the notes.
- You will need *at least two instances* of these data structures in order to solve this problem, but you must decide whether you need two instances of the same data structure or one instance of two different data structures.
- Make sure that you keep track of when the hangar or waiting runway are full. And remember that you should never access the inner parts of a data structure directly!

Example 1

input	corresponding **output**

`Brightedge no`
`Freebullet no`
`Flightcomet no`
`Dreamcharm yes`
`Nimblecomet yes`
`Sweet Robin no`
`Numb Viper yes`
`Nervous Eagle yes`
`Mellow Ghost no`
`Ragged Queen no`
	`Brightedge`
	`Freebullet`
	`Flightcomet`
	`Sweet Robin`
	`Mellow Ghost`
	`Ragged Queen`
	`Nervous Eagle`
	`Numb Viper`
	`Nimblecomet`
	`Dreamcharm`

Example 2

input	corresponding **output**
`Plane 2C no`	`Plane 2C`
`Supermarine Spitfire yes`	`Lockheed SR-71 Blackbird`
`Lockheed SR-71 Blackbird no`	`Max Plane`
`Max Plane no`	`Planey McPlaneface`
`Boeing 787 yes`	`Learjet 23`
`Cirrus SR22 yes`	`NCC-1701-D`
`Planey McPlaneface no`	`Cirrus SR22`
`NCC-1701-D yes`	`Boeing 787`
`Learjet 23 yes`	`Supermarine Spitfire`
`Lockheed C-130 yes`	`Eyre Forzwan`
`Planespotting yes`	`Avro Vanquish`
`Eyre Forzwan no`	`Apollo 12`
`Avro Vanquish no`	`Duck Duck Goose`
`Apollo 12 no`	`Red Baron`
`Duck Duck Goose no`	`Douglas DC-3`
`Swordfish II yes`	`Bleriot XI`
`Red Baron no`	`Cessna 172`
`Douglas DC-3 no`	`Swordfish II`
`Cessna 172 yes`	`Planespotting`
`Bleriot XI yes`	`Lockheed C-130`

Themis

Please download `lab1planes.zip` and write your own solution into `lab1planes.py`. This is also the only file you need to upload. The following files are included by Themis, you should *not* upload them and you should not have any conflicting definitions in your own file: `linkedlist.py`, `queue.py`, and `stack.py`.