Permission Dependency Resolver

Your task is to write a class that will resolve dependencies for user permissions. For example, if you wish to grant a user the `edit`, permission, they need to already have the `view` permission. In this challenge, that would be encoded as follows:

```
{
  "view" => [],
  "edit" => ["view"]
}
```

This means that the `view` permission has no dependencies (hence the empty array), whereas the `edit` permission is dependent on the `view` permission, and can't be added to a user unless `view` is present.

The class you write must take an object like the one shown above as the only argument in its constructor. The class must implement the following three functions:

`can_grant?(existing, perm_to_be_granted)`

This function determines whether a permission can be granted, given that a user already has existing permissions. The parameter `existing` will be an array of strings representing permissions already granted, and `perm_to_be_granted` is a string representing the permission we wish to grant. It should return `true` if the permission can be granted and `false` if not.

`can_deny?(existing, perm_to_be_denied)`

This function whether a permission can be denied (removed), and still satisfy all dependencies of the other permissions. For example if a user has `view` and `edit` permissions, it should not be possible to remove the `view` permission since `edit` depends on it. The parameter `existing` will be an array of strings representing permissions already granted, and `perm_to_be_denied` is a string representing the permission we wish to deny. It should return `true` if the permission can be denied and `false` if not.

`sort(permissions)`

This function should sort a given list of permissions into the order that they should be granted to satisfy dependencies. For example, if we sort `view` and `edit` permissions, `view` must come before `edit`, since `edit` depends on `view` already being granted. The parameter permissions is an array of strings representing the permissions to sort. The function should return an array of the given permissions in dependency order.

To run the specs, run `bundle install`, and then run `rspec` in the project directory.