Here are some small exercises which you might find useful. These are not assessed, so you may discuss them freely with other students, demonstrators, and on the Forum. I have tried to create exercises which explore some important principles - both for object-oriented programming in general, and for the second assignment in particular. Some of these concepts are covered in the second lecture. There is not necessarily a single, correct answer - they are intended to stimulate discussion - so if you think you know the “answer”, it is still worthwhile discussing possible alternative solutions with other people.

Directories

The Java HashMap provides a convenient way of representing a “directory” which has unique keys. For example, a telephone directory...

```java
HashMap<String, String> directory = new HashMap<String, String>);
directory.put(name, number);
number = directory.get(name);
```

1 How would you create a Directory class which allowed you to look up entries, based on either the name, or the number? Think about some different ways of implementing this, and their relative advantages/disadvantages.

```java
Directory directory = new Directory();
directory.put(name, number);
number = directory.getByName(name);
name = directory.getByNumber(number);
```

2 A real directory is likely to contain multiple people with the same name (but different numbers). So, assuming that the names are not unique, how would you change the interface to support this? What type of object should `getByName()` return? How would you implement this?

Seasons

Assume that we are going to create a class to represent the seasons, with the following interface:

```java
public class Season {
    public static Season Spring();
    public static Season Summer();
    public static Season Autumn();
    public static Season Winter();
    public Season next();
    public void print();
}
```

3 Pick one way of implementing this with an emphasis on readability.

4 What do you think “new Season()” should do?

1 Assuming for now that the names are unique.
[5] What would you need to do to ensure that the following two values are “equal”?

```java
Season a = Spring().Next();
Season b = Spring().Next();
```

[6] Extend your class to include an AverageTemperature for each Season. What would you expect the following to do? What does your implementation do?

```java
Season a = Spring();
Season b = Spring();
a.SetAverageTemperature(30);
print b.AverageTemperature();
```