## 给出一个Java ADT题目

**某公司拟设计和开发一个停车场管理系统，其基本需求陈述如下：**

1. 一个停车场有n个车位(n>=5)，不同停车场包含的车位数目不同。
2. 一辆车进入停车场，如果该停车场有空车位且其宽度足以容纳车的宽度，则可以在此停车。支持自动分配车位和指定停车位两种分配方式。
3. 停在停车场里的车，随时可以驶离停车场。
4. 停车场管理员可以随时查看停车场的当前占用情况。

下图给出了一个包含13个停车位的小型停车场示例图，其中1-8号停车位较窄，9-13号停车位较宽。在当前状态下，第1、3、7、9、10号车位被占用，其他车位空闲。

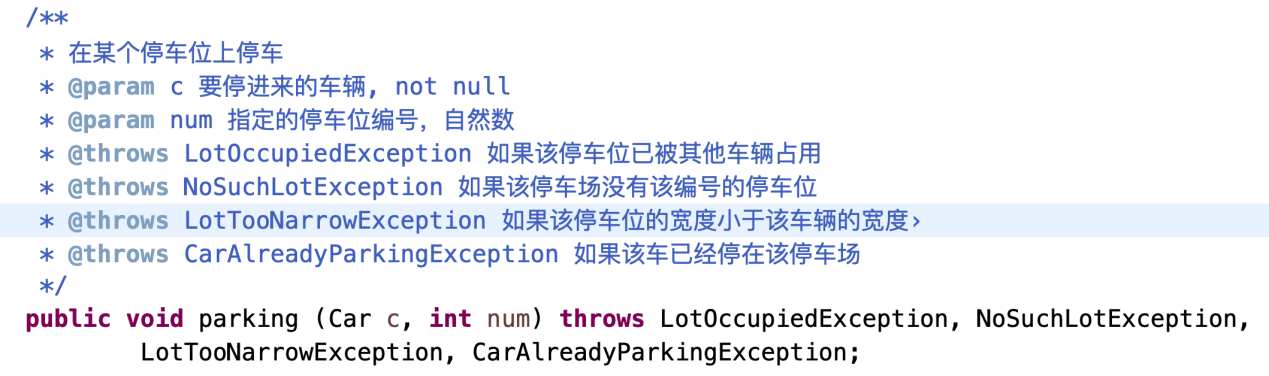


## 设计ParkingField接口，撰写spec

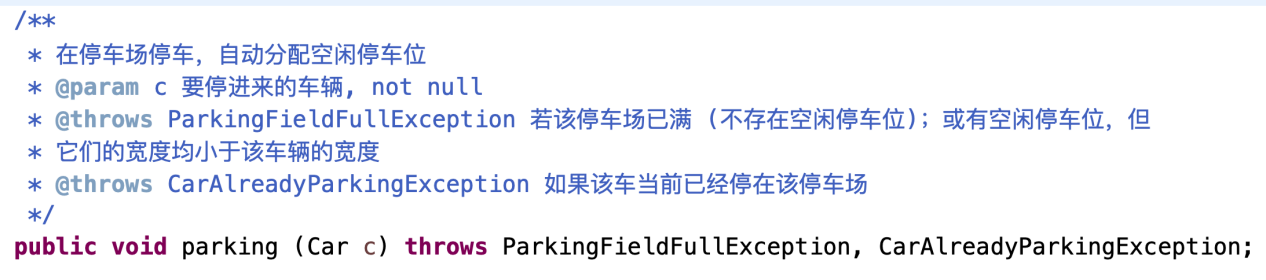
方法设计，撰写spec。注：根据1中的需求，给出方法的设计即可。异常处理这部分建议可以先保留，后面对应章节再处理。

针对停车需求，设计两种parking策略。

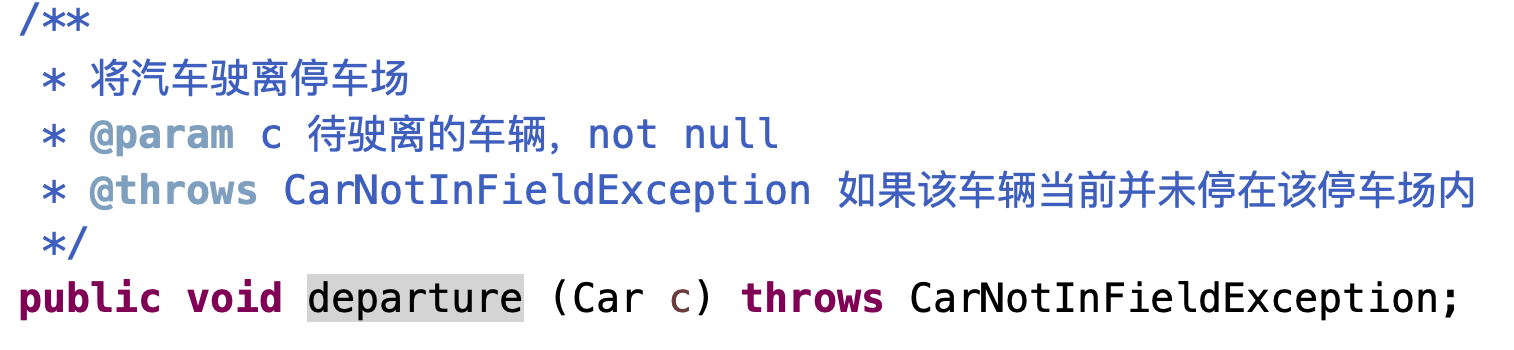
第1种：指定停车位



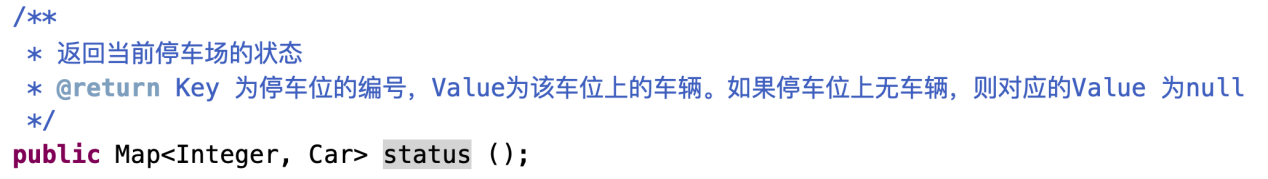
第2种：自动分配停车位



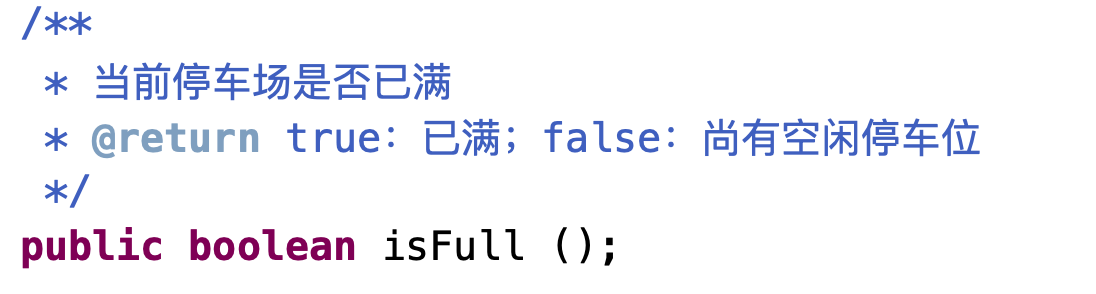
驶离停车场



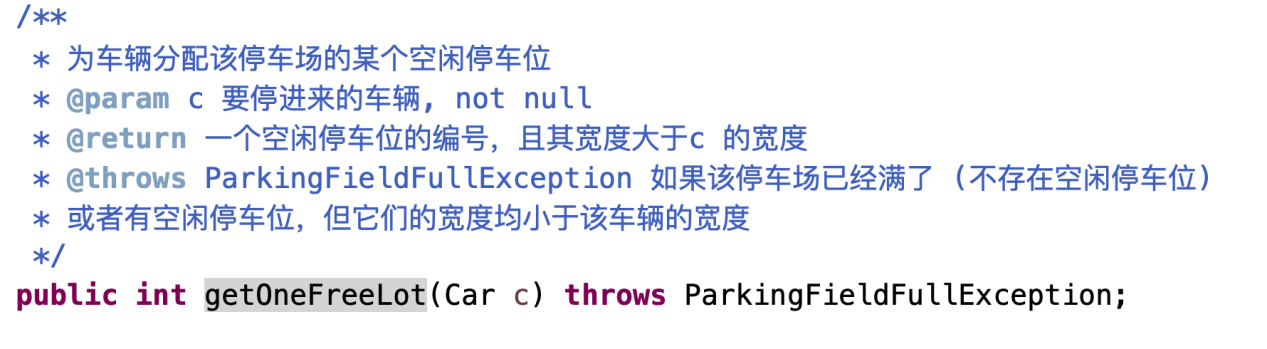
返回停车场停车状态



判断停车位是否已满

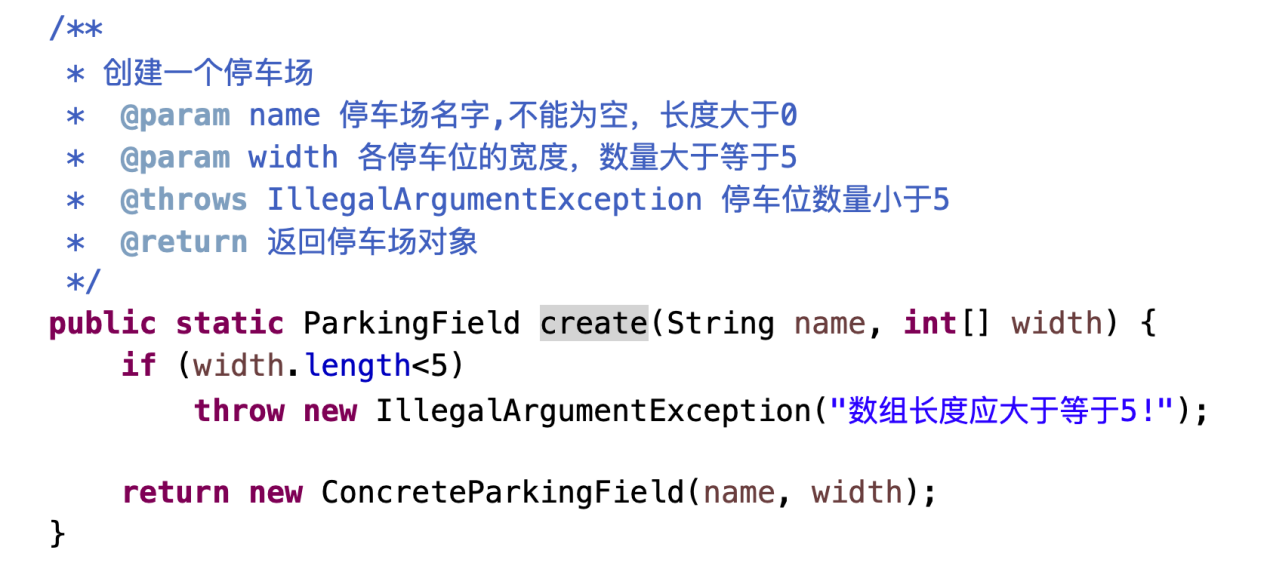


得到一个空闲停车位



## 为ParkingField接口增加静态工厂方法

增加静态工厂方法，创建实例



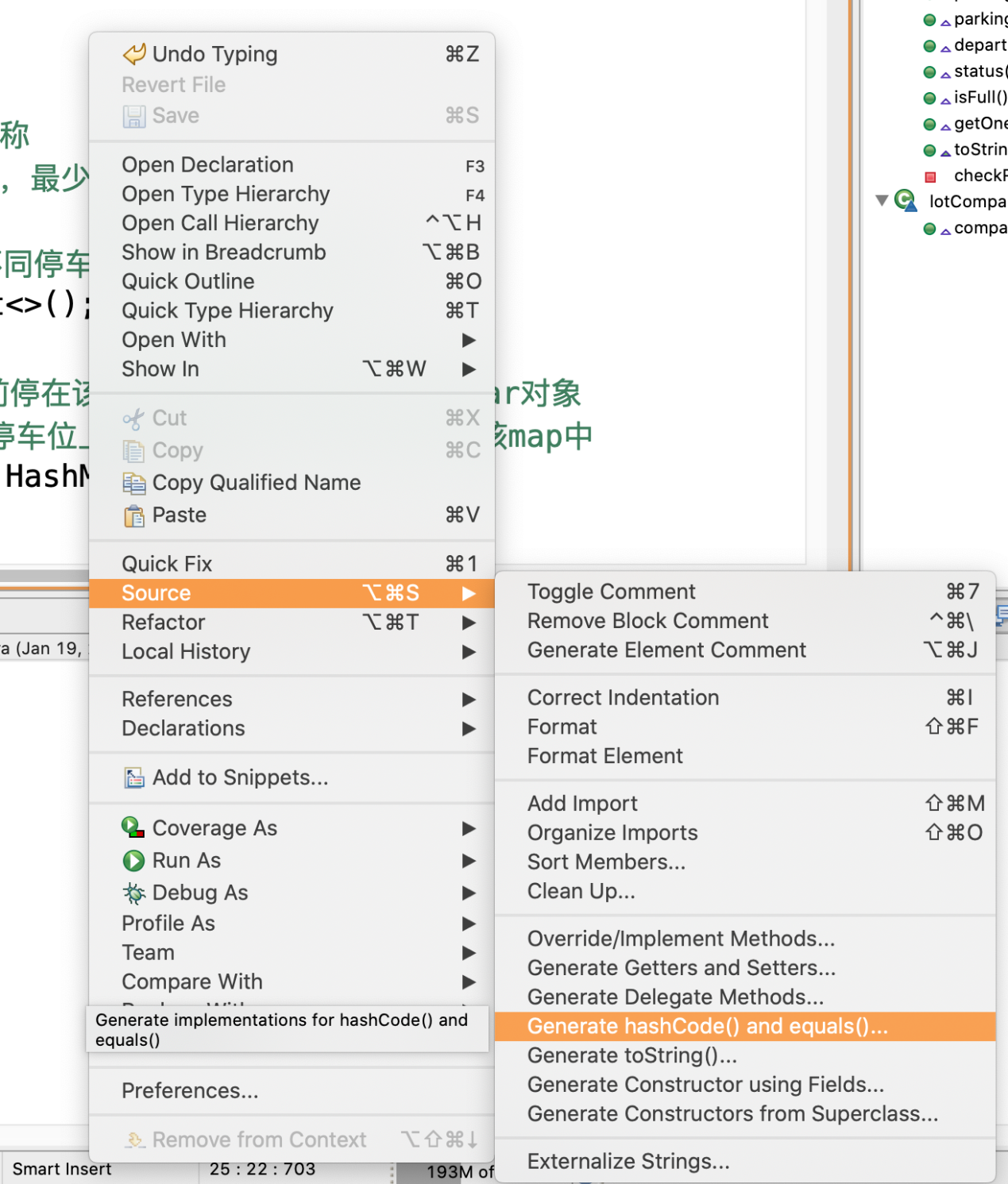
## 针对ParkingField接口设计测试用例

详见代码 TestParkingField类

## 编写具体实现类ConcreteParkingField

具体实现见代码ConcreteParkingField

此外：可介绍下equals()、hashCode()、toString() 通过IDE自动生成的方式。



设计rep

设计AF、RI

实现checkRep()

实现构造函数

实现接口方法

override equals()、hashCode()、toString() （ConcreteParkingField是mutable的，所以不需要重写equals()和 hashCode()）

(1) 停车场的名字和停车位数目

(2) 停车场车位的占用比例

(3) 每个停车位的编号、宽度、状态

例子：某ConcreteParkingField对象的名字是“A1”，它有5个停车位，状态如下表所示。

|  |  |  |
| --- | --- | --- |
| 停车位编号 | 停车位宽度 | 当前所停车辆 |
| 1 | 200 | 车牌号：AB001，宽度180 |
| 2 | 180 | 空闲 |
| 3 | 200 | 车牌号：CD002，宽度180 |
| 4 | 170 | 车牌号：EF003，宽度160 |
| 5 | 190 | 空闲 |

从上表看出，该停车场目前有60%的停车位已停车 (=3/5)。针对该例子，你所写的toString()的输出结果应为：

Parking field A1 has total number of lots: 5

Now 60% lots are occupied

Lot 1 (200): Car AB001

Lot 2 (180): Free

Lot 3 (200): Car CD002

Lot 4 (170): Car EF003

Lot 5 (190): Free

## 编写辅助类Car、Lot

详见Car和Lot两个类

Immutable类

设计rep，是否有多种表示？

设计RI，与Rep密切相关

实现checkRep()

设计方法

override equals()、hashCode()、toString()

## 编写客户端程序

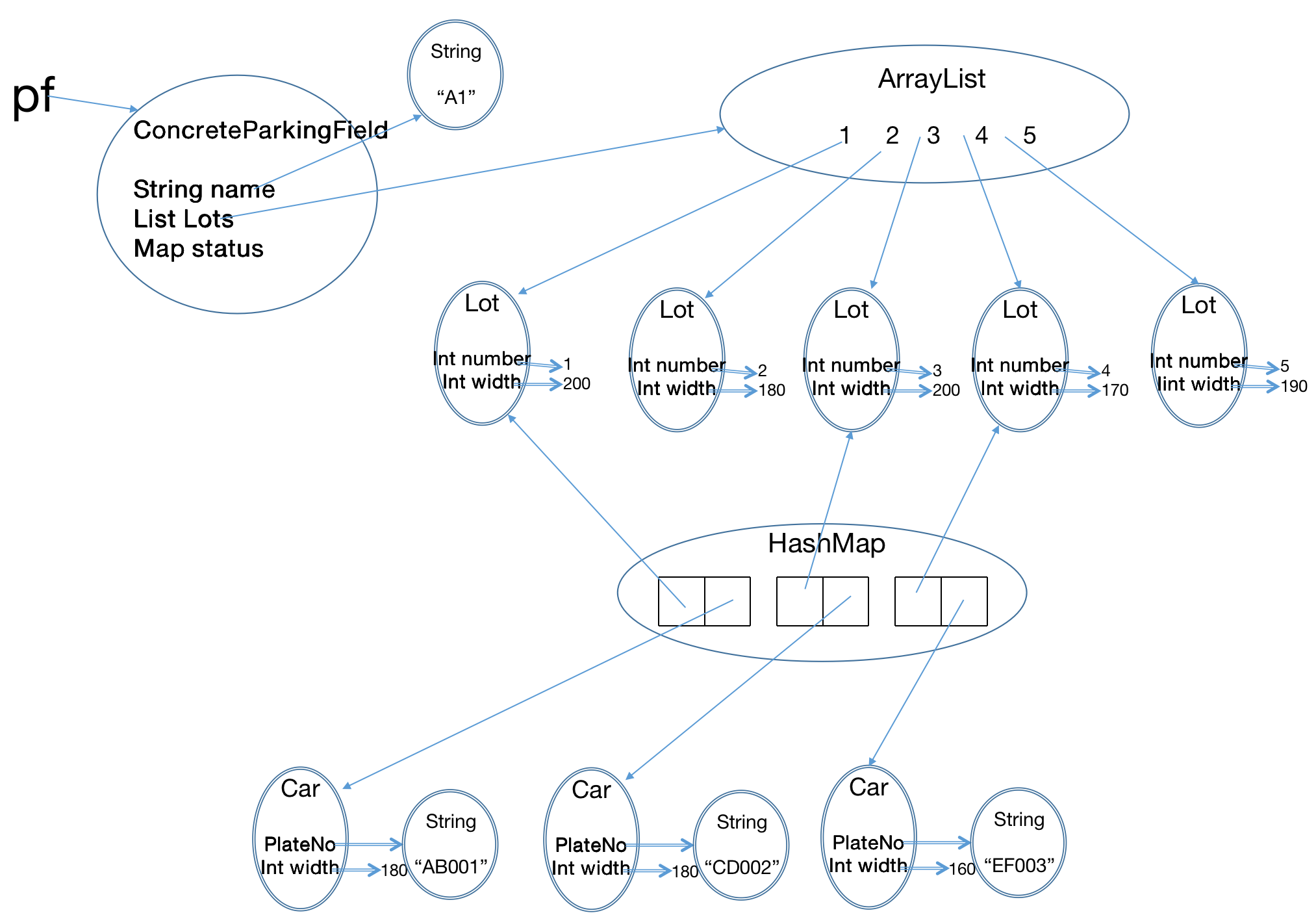
见Client，运行后按照提示操作即可。

某个停车场管理系统，实现一系列功能

## 绘制Snapshot Diagram

针对客户端程序运行到某个时刻，想象内存状态，绘制snapshot

针对5中toString的例子，状态图如下



## 表示泄露与安全性

考察ParkingField、Car、Lot这些ADT，考虑以下的客户端代码，判断这些ADT是否存在表示泄露？请逐项列出可能存在表示泄露的地方，分析其潜在风险，并给出其修改策略。

给出几个表示泄露的客户端例子，以及如何通过防御式拷贝策略修改ADT设计。

表示泄露的例子：

public Set<Lot> getLots(){

return this.lots; //存在表示泄露

//return Collections.unmodifiableSet(lots);

}

Status()方法因为返回的是Map<Integer, Car>，是新建的Map返回，所以不存在泄露(防御式拷贝)。可以同学生讲下

## 执行JUnit测试

以及查看测试覆盖度Eclemma

