Project – Car parking

Rules

1. You must solve the problem in Python without using any external libraries. Your solution must build and run on Linux.
2. Unit tests *are mandatory*, so please include tests/specs. Additionally, it's a huge plus if you test drive your code.
3. Please ensure that you follow the syntax and formatting of both the input and output samples. We validate submissions using automated tests.
4. We are really interested in your object orientated development skills, so please solve the problem keeping this in mind.
5. Please ensure that the coding conventions, directory structure and build approach of your project follow the conventions set by popular open source projects in the language that you're using.
6. When implementing this solution, please use Git for version control. We expect you to send us a zip/tarball of your source code when you're done that includes Git metadata (the .git folder) in the tarball so we can look at your commit logs and understand how your solution evolved.
7. *Please do not* make either your solution or this problem statement publicly available by, for example, using github or bitbucket or by posting this problem to a blog or forum.

Problem Statement

I own a multi-storey parking lot that can hold up to 'n' cars at any given point in time. Each slot is given a number starting at 1 increasing with increasing distance from the entry point in steps of one. I want to create an automated ticketing system that allows my customers to use my parking lot without human intervention.

When a car enters my parking lot, I want to have a ticket issued to the driver. The ticket issuing process includes us documenting the registration number (number plate) and the colour of the car and allocating an available parking slot to the car before actually handing over a ticket to the driver (we assume that our customers are nice enough to always park in the slots allocated to them). The customer should be allocated a parking slot which is nearest to the entry. At the exit, the customer returns the ticket which then marks the slot they were using as being available.

Due to government regulation, the system should provide me with the ability to find out:

a) Registration numbers of all cars of a colour.
b) Slot number in which a car with a given registration number is parked.
c) Slot numbers of all slots where a car of a colour is parked.

We interact with the system via a simple set of commands which produce a specific output. Please look at the example below, which includes all the commands you need to support - they're self-explanatory. The system

should allow input in two ways. Just to clarify, the same codebase should support both modes of input - we don't want two distinct submissions.

1) It should accept a filename as a parameter at the command prompt and read the commands from that file
2) It should provide us with an interactive command prompt-based shell where commands can be typed in

Examples

Example1: Using an input file

This example will take input from the input file (file_inputs.txt) and write the output in (file_output.txt). This is without interactive mode.

Command to run:

```
$ my_program file_inputs.txt > file_output.txt
```

Sample Input file (file_inputs.txt)

```
create_parking_lot 6
park KA-01-HH-1234 White
park KA-01-HH-9999 White
park KA-01-BB-0001 Black
park KA-01-HH-7777 Red
park KA-01-HH-2701 Blue
park KA-01-HH-3141 Black
leave 4
status
park KA-01-P-333 White
park DL-12-AA-9999 White
registration_numbers_for_cars_with_colour White
slot_numbers_for_cars_with_colour White
slot_number_for_registration_number KA-01-HH-3141
slot_number_for_registration_number MH-04-AY-1111
```

Expected Output (to console, newline after every output)- file_output.txt:

```
Created a parking lot with 6 slots
Allocated slot number: 1
Allocated slot number: 2
Allocated slot number: 3
Allocated slot number: 4
Allocated slot number: 5
Allocated slot number: 6
Slot number 4 is free
Slot No.    Registration No    Colour
1    KA-01-HH-1234    White
2    KA-01-HH-9999    White
3    KA-01-BB-0001    Black
5    KA-01-HH-2701    Blue
6    KA-01-HH-3141    Black
Allocated slot number: 4
Sorry, parking lot is full
KA-01-HH-1234, KA-01-HH-9999, KA-01-P-333
1, 2, 4
```

```
6
Not found
```

Example2: Interactive mode

This example will take input in interactive mode (through console) and give immediate output.

Command to run:

To run the program and launch the shell:

```
$ my_program
```

Input/output commands

Assuming a parking lot with 6 slots, the following commands should be run in sequence by typing them in at a prompt and should produce output as described below the command:

Input:

```
create_parking_lot 6
```

Output:

```
Created a parking lot with 6 slots
```

Input:

```
park KA-01-HH-1234 White
```

Output:

```
Allocated slot number: 1
```

Input:

```
park KA-01-HH-9999 White
```

Output:

```
Allocated slot number: 2
```

Input:

```
park KA-01-BB-0001 Black
```

Output:

```
Allocated slot number: 3
```

Input:

```
park KA-01-HH-7777 Red
```

Output:

```
Allocated slot number: 4
```

Input:

```
park KA-01-HH-2701 Blue
```

Output:

```
Allocated slot number: 5
```

Input:

```
park KA-01-HH-3141 Black
```

Output:

```
Allocated slot number: 6
```

Input:

```
leave 4
```

Output:

```
Slot number 4 is free
```

Input:

```
status
```

Output

```
Slot No.
Registration No
```

```
Colour
1
KA-01-HH-1234
White
2
KA-01-HH-9999
White
3
KA-01-BB-0001
Black
5
KA-01-HH-2701
Blue
6
KA-01-HH-3141
Black
```

Input:

```
park KA-01-P-333 White
```

Output:

```
Allocated slot number: 4
```

Input:

```
park DL-12-AA-9999 White
```

Output:

```
Sorry, parking lot is full
```

Input:

```
registration_numbers_for_cars_with_colour White
```

Output:

```
KA-01-HH-1234, KA-01-HH-9999, KA-01-P-333
```

Input:

```
slot_numbers_for_cars_with_colour White
```

Output:

```
1, 2, 4
```

Input:

```
slot_number_for_registration_number KA-01-HH-3141
```

Output:

```
6
```

Input:

```
slot_number_for_registration_number MH-04-AY-1111
```

Output:

```
Not found
```

Record this project in GitHub

We Want you to commit your progress to GitHub, with proper commit message.
Share the link to the project for review.