**Problem Statement**

We own a courier facility which segregate the courier for different cities into different racks that can hold up to 'n' packages at any given point in time. Each level in the rack has 2 slots. The slots is given a number first to all in one column starting at 1 increasing with increasing level from the entry point in steps of one till n/2 and then to the second column as n/2+1 at the of the topmost level decreaseing by 1 at each lower level. We want to create an automated ticketing system that allows my facility man to use my alloted slot without any error.

Eg: if I have a rake with 10 slots, the arrangement will be like:

|  |  |
| --- | --- |
| 5 | 6 |
| 4 | 7 |
| 3 | 8 |
| 2 | 9 |
| 1 | 10 |

When a parcel enters my racks, we want to have a ticket issued to the parcel.

The ticket issuing process includes us documenting the parcel code and

the weight(in gm) of the parcel and allocating an available slot to the parcel before actually handing over a ticket to the facility man

(we assume that our facility men are nice enough to always park in the slots allocated to them). The Parcel should be allocated a 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) Parcel codes of all parcels of a particular weight.

b) Slot number in which a parcel with a given code is parked.

c) Slot numbers of all slots where a parcel of a particular weight is parked.

We interact with the system via a simple set of commands which produce a specific output. Please take a 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 provide us with an interactive command prompt based shell where commands can be typed in

2) It should accept a filename as a parameter at the command prompt and read the commands from that file

\*Example: File\*

To run the program:

$ my\_program file\_inputs.txt > output.txt

**Input (in file):**

create\_parcel\_slot\_lot 6

park 1234 400 park 9999 400 park 0001 600 park 7777 100 park 2701 700 park 3141 600

leave 5 for delivery status

park 333 400

park 9999 400 parcel\_code\_for\_parcels\_with\_weight 400 slot\_numbers\_for\_parcels\_with\_weight 400 slot\_number\_for\_registration\_number 3141 slot\_number\_for\_registration\_number 1111

**Output (to console, newline after every output):**

Created a parcel slot with 6 slots

Allocated slot number: 1

Allocated slot number: 6

Allocated slot number: 2

Allocated slot number: 5

Allocated slot number: 3

Allocated slot number: 4

Slot number 5 is free

Slot No. Registration No. Weight

|  |  |
| --- | --- |
| 1 1234 | 400 |
| 6 9999 | 400 |
| 6 0001 | 600 |
| 3 2701 | 700 |
| 4 3141 | 600 |

Allocated slot number: 5

Sorry, parcel slot is full

1234, 9999, 333

1, 6, 5

4

Not found

**\*Example: Interactive\***

To run the program and launch the shell:

$ my\_program

Assuming a parcel slot 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 parcel slot with 6 slots

Input:

○ park 1234 400

Output:

○ Allocated slot number: 1

Input:

○ park 9999 400

Output:

○ Allocated slot number: 6

Input:

park 0001 600

Output:

Allocated slot number: 2

Input:

park 7777 100

Output:

Allocated slot number: 5

Input:

park 2701 700

Output:

Allocated slot number: 3

Input:

park 3141 600

Output:

Allocated slot number: 4

Input:

leave 5 for delivery

Output:

Slot number 5 is free

Input:

status

Output (we've used a table to make our lives easier, but tab delimited output is fine):

Slot No Registration No Weight

1 1234 400

6 9999 400

2 0001 600

3 2701 700

4 3141 600

Input:

park 333 400

Output:

Allocated slot number: 5

Input:

park 9999 400

Output:

Sorry,parcel slot is full

Input:

parcel\_code\_for\_parcels\_with\_weight 400

Output:

1234, 9999, 333

Input:

slot\_numbers\_for\_parcels\_with\_weight 400

Output:

1, 6, 5

Input:

slot\_number\_for\_registration\_number 3141

Output:

4

Input:

slot\_number\_for\_registration\_number 1111

Output:

Not found