Parking System

Problem Description:

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

When a car enters the parking lot, we want to have a ticket issued to the driver. The ticket issuing process includes:-

1. We are taking note of the number written on the vehicle registration plate and the age of the driver of the car.
2. And we are allocating an available parking slot to the car before actually handing over a ticket to the driver (we assume that our customers are kind enough to always park in the slots allocated to them).

The customer should be allocated a parking slot that is nearest to the entry. At the exit, the customer returns the ticket, marking the slot they were using as being available.

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

- Vehicle Registration numbers for all cars which are parked by the driver of a certain age.
- Slot number in which a car with a given vehicle registration plate is parked.
- Slot numbers of all slots where cars of drivers of a particular age are parked.

Requirements:

- **Python 3.7.3**(Preferrable) or above **Python 3.5**
- **Windows or Ubuntu**

Setup:

1. Install Python that satisfies the above requirements (3.7.3 or above python 3.5)

   Windows:

   1. https://www.python.org/downloads/release/python-373/ go to this link and you can choose Windows x86-64 executable installer or your choice to download
   2. Execute the downloaded file to install python
   3. Add python path in windows. Refer to this link for further information. https://datatofish.com/add-python-to-windows-path/

   Ubuntu:

   Execute the commands in the terminal.

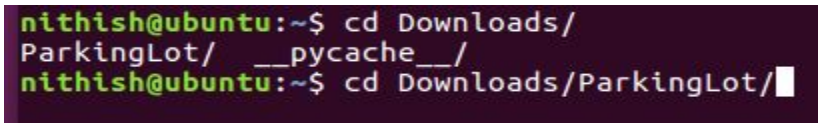
   1. sudo apt update
   2. sudo apt install software-properties-common
   3. sudo add-apt-repository ppa:deadsnakes/ppa
   4. sudo apt install python3.7

   For more details refer this link
   https://linuxize.com/post/how-to-install-python-3-7-on-ubuntu-18-04/

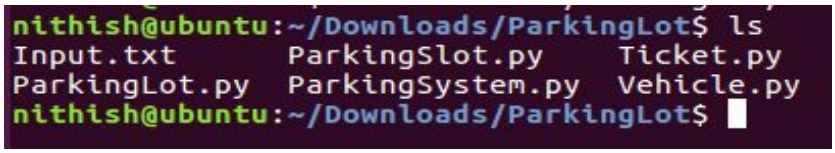
2. Check enum library is already there if it is not, Execute the command in terminal
   - pip install enum

Instructions to Execute:

1. Download the ParkingLot project
2. Open the command prompt or terminal
3. Point to the project location by using the cd command. In my case, it is in the Downloads/ParkingLot path



4. Check all the files are there



5. Run the following command on the terminal. ParkingSystem.py is the main file. You should definitely give the input file name on the command. Please check Is the file name is correct while executing the command.

   Command Format:

   python ParkingSystem.py <Input FIle Name> or
   python3 ParkingSystem.py <Input File Name>

   - Replace <Input File Name> in your corresponding input file name. But it should be in the same location where all the files are located. For **Example,**

   ```
   python3 ParkingSystem.py Input.txt
   ```

   Command to Execute:

   python ParkingSystem.py Input.txt or
   Python3 ParkingSystem.py Input.txt

Data Structures Used:

- A **list** is a data structure in Python that is a mutable, or changeable, ordered sequence of elements. Each element or value that is inside of a list is called an item. Just as strings are defined as characters between quotes, lists are defined by having values between square brackets [ ].

- **Dictionary** in Python is an unordered collection of data values, used to store data values like a map, which unlike other Data Types that hold only a single value as an element, Dictionary holds key:value pair. Key value is provided in the dictionary to make it more optimized.

Input File Description:

Command	Description of Command
Create_parking_lot 6	Create a parking lot of 6 slots
Park KA-01-HH-1234 driver_age 21	Park car with vehicle registration number " KA-01-HH-1234", and the vehicle is driven by the driver of age 21
Park PB-01-HH-1234 driver_age 21	Park car with vehicle registration number " PB-01-HH-1234", and the car is driven by the driver of age 21
Slot_numbers_for_driver_of_age 21	Return all Slot Number(Comma-separated) of all cars which have drivers with age==21
Park PB-01-TG-2341 driver_age 40	Park car with vehicle registration number " PB-01-TG-2341", and the car is driven by the driver of age 40
Slot_number_for_car_with_number PB-01-HH-1234	Return slot number for the car with registration number "PB-01-HH-1234"
Leave 2	Vacate the slot number 2 from the parking lot, i.e. car which was parked at slot number 2 has left the space if there exists no car at slot number 2, print "Slot already vacant"
Park HR-29-TG-3098 driver_age 39	Park car with vehicle registration number " HR-29-TG-3098", and the car is driven by the driver of age 39
Vehicle_registration_number_for_driver_of_age 18	Get all parked vehicle registration number of cars parked by the driver of age 18

Input:

```
Create_parking_lot 6
Park KA-01-HH-1234 driver_age 21
Park PB-01-HH-1234 driver_age 21
Slot_numbers_for_driver_of_age 21
Park PB-01-TG-2341 driver_age 40
Slot_number_for_car_with_number PB-01-HH-1234
Leave 2
Park HR-29-TG-3098 driver_age 39
Vehicle_registration_number_for_driver_of_age 18
```

Output:

1. Created a parking lot with 6 slots
2. Car with vehicle registration number "KA-01-HH-1234" has been parked at slot number 1
3. Car with vehicle registration number "PB-01-HH-1234" has been parked at slot number 2
4. These are the slot numbers found for this particular age 1, 2
5. Car with vehicle registration number "PB-01-TG-2341" has been parked at slot number 3
6. The slot number for the vehicle registration number is 2
7. Slot number 2 vacated, the car with vehicle registration number "PB-01-HH-1234" left the space, the driver of the car was of age 21
8. Car with vehicle registration number "HR-29-TG-3098" has been parked at slot number 2
9. There is no vehicle registration number found for this particular age

Class Diagram: