PARKING LOT SIMULATOR

(Python project)

By : Gayathri Chennakrishnam
Batch : Subramanyam

TABLE OF CONTENTS:

Introduction :

This is a Python based project for implementing Parking lot simulator which tickets vehicles(Cars in this instance) into parking slots without human interface for efficient management of parking lot.

Python files and other files :

This project includes following python files:

Parking_Lot.py : This is the main file that runs the app in order to obtain the required output. It contains the main class (Parking), that has all the main function methods such as park, leave , status etc.

Vehicle.py : This file creates a vehicle object and in-turn a Car instance which is used in the app. It includes the characteristics of the Car like Registration number and color of the vehicles to monitor and manage the lot.

test_parking_lot.py : This file is used to unit test the app for optimisation and expected output.

This project includes following other files such as run_test_case.txt which contains multiple input commands that can be given as input directly instead of individual commands.

Key Features :

*Inter-active command prompt for learning the constant status of the parking lot.

*extracting data based on different factors like Parking slots sorted by registration numbers and or colours of the vehicles.

*As mentioned above, one of the key features is the option of inputting multiple commands as a file.

Technologies Used :

*Python 3.9 Version was used for the core code source.

*Flake 8 was used for clean and modular code.

*As well as few extensions such as Python Indent, Trailing Spaces etc.

Imports Used :

*argparse : Command-line parsing library

This module is an opt-parse-inspired command-line parsing library that:

- handles both optional and positional arguments
- produces highly informative usage messages
- supports parsers that dispatch to sub-parsers

*sys : module sys

This module provides access to some objects used or maintained by the interpreter and to functions that interact strongly with the interpreter.

*unittest : module unittest

Python unit testing framework, based on Erich Gamma's JUnit and Kent Beck's Smalltalk testing framework (used with permission).

This module contains the core framework classes that form the basis of specific test cases and suites (TestCase, TestSuite etc.), and also a text-based utility class for running the tests and reporting the results.

Future Aspects :

*Would like to enhance the User Interactive aspect of the app by implementing visual rich graphical-user-interface modules.

*Would like to implement different “levels/floors” parking slots management.

*Could make the app more real-time to obtain real-time data.

*Could implement the core idea for multiple vehicles(two-wheelers,four-wheelers etc.)parking slots management.

References:

Instructor/mentor sessions
Google
StackOverFlow
StackExchange etc.

Thank you.