ICS 104 LAB – T212 – SEC 52/54 (FNb) – Lab Project

KFUPM Parking Management System

A. Description

Each team is expected to deliver a simple python implementation of KFUPM Parking System where a user can do many things in the system. For example, he can park his vehicle (check-in), exit (check-out) a lot, view current parked vehicle, register his vehicle and update some data. You need three text files for the system as follows.

1. The information of parked vehicles are stored in "**parkedVehicles.txt**" file, such that each line represents a record that corresponds to a unique parked vehicles in a parking building after successful check-ins. Each record contains the following information seperated by formatted spaces:

- Vehicle Plate Number (unique 7 characters: 4 digits followed by 3 letters)
- KFUPM ID
- Building Number
- Check-In Time

2. The information of parking buildings are stored in "**buildings.txt**" file, such that each line represents a record that corresponds to a unique parking building. Each record contains the following information seperated by formatted spaces:

- Building No (unique 3 digits)
- Department Name
- Parking Capacity
- No. of Parked Vehicles

3. The information of registered vehicles are stored in "**registeredVehicles.txt**" file, such that each line represents a record that corresponds to a unique registered vehicle. Each record contains the following information seperated by formatted spaces:

- Vehicle Plate Number (unique 7 characters: 4 digits followed by 3 letters)
- KFUPM ID
- Iqama
- Vehicle Model
- Manufacture Year
- Color

You are required to develop a menu-driven program that displays the following menu and keeps displaying it until a user chooses the exit option.

```
KFUPM Parking Management System
==============================
1. Register a Vehicle
2. View Registered Vehicles
3. View Parking Buildings
4. Check-in for Parking
5. View Parked Vehicles
6. Search a Vehicle
7. Check-out Parking
8. Update a Parking Building
9. Print Parking Statistics
10. Exit
===============================
Enter your choice index (?):
```

Below are the details for each possible choice:

1. **Register a Vehicle:**
   The user should enter all required information as per the record items (see above)
   You should validate the user input as follows:
   - *Vehicle Plate Number*: it should be a string of length 7, of which there should be exactly 4 digits followed by 3 letters. The plate number should not match any of existing plate numbers.
   - *KFUPM ID:* It should be a string of length 10, and it should not match any existing KFUPM ID.
   - *Iqama:* 10 digits iqama number
   - *Model :* Vehicle model name, string and cannot be null. (e.g. "Toyota Yaris")
   - *Year* : Manufacture Year, should be an integer of 4 digits length
   - *Color :* you can accept any input string, but should ignore any leading or tailing spaces

2. **View Registered Vehicles:**
   The user should be able to view all the registered Vehicles. For each vehicle, you shall print the following information in a formatted table:
   - Vehicle Plate Number
   - KFUPM ID
   - Vehicle Model
   - Manufacture Year
   - Vehicle Color

3. **View Parking Buildings:**
   The user should be able to view all the buildings reserved for vehicle parking. For each building, you shall print the following information in a formatted table:
   - Building No
   - Department Name
   - Parking Capacity
   - No. of Parked Vehicles

4. **Check-in for Parking:**
   The user should be able to check-in her vehicle for parking by entering all required information as per the record items (see above)
   You should validate the user input as follows:
   - *Vehicle Plate Number:* It should match an entry in the registered vehicles file.
   - *KFUPM ID* :It should be a string of length 10, and it should match an existing KFUPM ID in the **registeredVehicles.txt** file.
   - *Building Number:* It should match an entry in the **buildings.txt** file.
   - *Check-In Date-Time:* python datetime generated by the program

   { **Hints**: python package you will need:
   `from datetime import datetime`
   Methods you will need:
     - `ts = datetime.timestamp(datetime.now())`
     - `datetime.fromtimestamp(ts)`

   }

5. **View Parked Vehicles:**
   The user should be able to view all the parked Vehicles in a particular building chosen by the user. When the user select this menu (05. View Parked Vehicles), the system prompts the user to enter a particular building number. For each parked vehicle, you shall print the following information in a formatted table:
   - Vehicle Plate Number
   - KFUPM ID
   - Check-In Date-Time: Shown as Date, Time (hours, mins)

6. **Search a Vehicle:**
   The user should be able to search a vehicle by its plate number, to view its details and registration status, and parking status. The information should include the following:
   - All Vehicle Details given during registraion
   - Parking Status: if it is parked inside the campus inside a building, then the system should mention building number, and check-in Time

7. **Check-out Parking:**
   The user should be able to check-out parking by removing corresponding vehicle entry from the **parkedVehicles.txt** file. Below are the steps:
   - System will ask the user to enter Vehicle plate number
   - The user will enter the Plate Number for check-out
   - Error messeage should be display, if the user entered a wrong plate number
   - Then the program will display the parking information
   - user needs to confirm checking-out the parking
   - if the user confirms, the corresponding record shall be deleted from the file

8. **Update a Parking Building:**
   - The user will enter a Building Number
   - Then the program will display the Building info in a proper format
   - The items that can be modified are: building name and parking capacity
   - The user should be able to modify one, two, or none of the above items.
   - File shall be updated accordingly
   - Note: Parking capacity: should be an int number greater than or equal to the number of already parked vehicles in that particular building

9. **Print Parking Statistics:**

   For each building, the program should print the parking information in a formatted table:

   - Building Number
   - Building Name
   - Total Parking Capacity, parked cars, and space available
   - List of all parked cars info
10. Exit: Exit the system

B. Deliverables:

Each team has to submit:

- The code as a Jupyter notebook
- The report as part of the Jupyter notebook or as a separate word file. The report will describe how they solved the problem. In addition, they need to describe the different functions with their task and screen shots of their running code.   (worth 5%)
- Comments are important they are worth. (worth   5%)
- The code must use meaningful variable names and modular programming (worth 10%). Evaluation criteria include the following:
  - Use meaningful variable names
  - Modularity: Your program must contain as many functions as needed. You need to divide your problem into small tasks and each task is handled by a separate function.
  - Global variables are not allowed. Students should learn how to pass parameters to functions and receive results.

C. Project Demo/presentation

- Each team is required to present his project:
- The week of May 8-12 will be used for lab project presentations. You need to arrange for your students presentations via MS Teams or physically.
- A slot of 15 minutes will be allocated to each team for their presentation and questions
- Students who do not appear for lab demo/presentation will get 0.
- During the demo, each member is expected to run the program and perform some of its functionalities.
- Team members should be ready to answer any question about their code. So, make sure you read and understand all parts of the code.
- Failing to answer questions means losing major part of the project grade.

20% of the grade are highlighted above (shaded blue).

The remaining 80% will be on the code itself and presentation (Please see below section)

D. Grading (80%)

Item	**Points**
Register a Vehicle	7
View Registered Vehicles	6
View Parking Buildings	6
Check-In for Parking	6
View Parked Vehicles	6
Search a Vehicle	6
Check-out Parking	6
Update a Parking Building	7
Print Parking Statistics	6
Exit	4
Demo Presentation Performance	20

E. Guidelines

- Students must submit a working program. Non-working parts can be submitted separately. If a team submits a non-working program, it loses 20% of the grade.
- User input must be validated by the program i.e. valid range and valid type
- The deadline for submitting the lab project is Friday May 6 before midnight.
- Submitting Saturday before midnight will lead to 5% penalty
- Submitting Sunday before midnight 15% penalty

GOOD LUCK