**M6Pro Contact Manager Application**

**Assignment Objective**:

Requires using lists and dictionaries, while also incorporating string methods, exception handling, and reading/writing to CSV files.

**Data Structure:**

Use a list to store multiple contacts. Each contact should be represented as a dictionary with the following keys: `FirstName`, ‘lastName`, `phone`, `email`.

**How the program should work?**

The program is menu driven. Menu option and their functionality is explained below

**1) Add Contact:**

Prompt the user to enter a new contact's details (First name, last name, phone and email). Validate input

* Check if phone is numeric. Must be entered in the following format “(XXX)XXX-XXXX”
* Check if email contains "@" and ".").
* Use string methods to format the name (e.g., capitalize the first letter).

**2) View Contacts:**

Display all contacts in a formatted manner (**Tabular format**, MUST have a header row(column titles))

**3) Search Contact:**

* Allow the user to search for a contact by **lastName**. Use string methods to ignore case.
* If contact not found , notify user

**4) Remove Contact:**

* Allow the user to remove a contact by BOTH firstName and lastName. So BOTH must be verified, ensure the right contact is found before attempting to delete it.
* If contact not found, notify user

**5) Update Contact:**

* Allow the user to update phone number for a contact by lastName. If multiple contacts have the same last name, display the contacts and ask the user to verify the one they want to change.
* Ensure the name is found before attempting to change the phone number.
* If contact not found, notify user

**CSV Handling:**

* Create/Implement function to load contacts from a CSV file into the list of dictionaries when the program starts ( **you will have to create a CSV file that references a minimum of 4 contacts**)
* Implement a function to save the contacts back to a CSV file when the program exits.

**Exception Handling:**

* Handle exceptions for file operations (e.g., file not found).
* Handle any potential value errors when processing input (e.g., if the user inputs an invalid phone number).
* Ensure that the program doesn’t crash due to unforeseen errors, and provide user-friendly error messages.

**Example Workflow:**

1. Load contacts from `contacts.csv` at startup.

2. Display a menu with options to add, view, search, remove contacts, update contacts and exit.

3. Prompt the user for their choice and execute the corresponding function.

4. Save contacts back to `contacts.csv` upon exit.

**Important Submission requirements:**

- finished code solution file(s) .2 files minimum

- docstring and pseudocode is **required**