Geography Challenge Game - In-Class Exercise

Develop an interactive memory game that test or improves user's knowledge of geography. The core game presents a country (or state) and asks the user to provide its capital. The program checks if the answer is correct and awards points accordingly. This is a **freeform assignment** - you have creative freedom to design what constitutes a "win" condition and how the game progresses.

Integrate as many concepts from the course (dictionaries, File IO, string and list manipulation, controls structures, functions) and good programming practices (use of constants, friendly user messaged, clean code, automated tests, systematic debugging) as possible.

Basic Requirements

- Present a country (or state) name to the user
- Accept user input for the capital city
- Check if the answer is correct
- Award points for correct answers
- Make answers **case-insensitive** (e.g., "Paris", "paris", and "PARIS" should all be accepted for France)
- Program is as modular as possible, meaning code for specific tasks should be in functions where applicable.

You decide end condition and/or winning condition for the game. Some ideas:

- User types quit
- Reach a certain number of points (e.g., 10 points)
- Answer all questions correctly
- Achieve a certain accuracy percentage (e.g., 80% correct)
- Complete a set number of rounds
- Or any other creative conditions you can think of!

Academic integrity

Your programs must be your work (or your pair's work, if working in a pair). You may use all permitted *learning* resources, but the code you submit should be written by you. Be prepared to explain the logic and functionality of your code.

Important: Using help sites such as StackOverflow or generative AI tools is not permitted. These resources can be helpful "once you know what you're doing," but they create a difficult grey area while learning. Instead, use Ed discussion, office hours, and talk (without sharing exact code) with other students.

Deliverable

Submit the working game and all necessary files to Gradescope. Submitted programs should work. Grading will be based on effort (any reasonable working game that reads its input data from file will receive full points).

Pair programming

You may work with a partner on this game following pair programming rules: both submit the same code (listing your partner's name in the header), and both receive the same grade. Work on one computer in the same physical location—splitting up work is not permitted. Switch `driver` (typing) and `observer` (reviewing) roles for every problem.

When submitting to Gradescope, indicate that it's a group submission.

A note on data

The lists of countries with capitals and states with capitals are provided for you. Country definitions can be a sensitive topic; different sources and governments recognize different sets of countries. The data we'll use in this exercise follows the official US recognition.

Background: Randomness

To make your game more engaging, you'll likely want to randomize the order of questions or select random countries to ask about. Python's `random` module provides useful functions for this purpose. Import it with `import random` at the top of your file. Key functions include `random.choice()` to select a random item from a list (e.g., `random.choice(countries)`), `random.shuffle()` to randomly reorder a list in place, and `random.randint(a, b)` to generate a random integer between `a` and `b` (inclusive). These functions are particularly useful for shuffling the order of questions, selecting random countries to quiz on, or generating multiple choice options.

Implementation Variations

Level 1: Basic (Hardcoded Data)

Store a few countries and capitals directly in your code (using dictionaries). Simple implementation is good for getting started and testing your game logic.

Example data structure:

```
countries = {
    "France": "Paris",
    "Japan": "Tokyo",
    "Brazil": "Brasilia"
}
```

Level 2: File-Based (Text File)

Store countries and capitals in a text file (format: one country per line, with country and capital separated by @ sign (we picked a separator that is not a part of any country's name)) and read the data from the file when the program starts.

Example file format (`countries.txt`):

```
France@Paris
Japan@Tokyo
Brazil@Brasilia
Germany@Berlin
```

Level 3: JSON File (States & Capitals)

Create a variation using US states and their capitals, where data is stored in a JSON file (JSON allows for more structured data and easier parsing).

Example JSON format (`states.json`):

```
{
    "California": "Sacramento",
    "Texas": "Austin",
    "New York": "Albany"
}
```

Level 4: Progress Tracking

Store results from previous rounds. Keep track of which capitals were answered correctly and don't ask questions about capitals that were previously answered correctly; this adds a "learning" element - the game gets easier as you improve!

Implementation ideas:

- Use a list or a new type called set to track correctly answered capitals
- Save progress to a file so it persists between game sessions
- Display progress statistics (e.g., "You've mastered 15 capitals so far!")

Level 5: Multiple Choice Mode

Instead of free-form text input, present the user with 4 capital city options, user selects one of the four choices (by number or letter). This is easier for users who aren't confident spellers but needs another step in implementation, generating wrong answers (distractors).

Example interaction:

```
What is the capital of France?
A) London
```

```
B) Paris
C) Madrid
D) Rome
Enter your choice (A/B/C/D):
```

More Challenges (Optional)

- **Timer mode**: Add a time limit for each question
- **Difficulty levels**: Easy (common countries), Medium, Hard (lesser-known capitals)
- **Hints system**: Provide hints after incorrect answers
- **Statistics**: Track accuracy, average response time, most difficult capitals
- **Shuffle questions**: Randomize the order of questions
- **Two-player mode**: Compete against a friend

Be creative, experiment with different approaches, and most importantly, have fun building your geography challenge game!