Lab

codeName: **Most Wanted**

Instructions

You have been contracted to build a prototype for a person search for a top secret government project. Your application will allow a user to search for a person, and then fetch various information about that person.

This project should be developed using the best practices of functional programming. This means avoiding global variables and side effects from your functions.

You have been given access to some starter code, including a sample dataset representing individuals. You are not required to use any of the starter code, but you must satisfy the requirements of the project. The project will be tested using data formatted according to the *data.js* file.

The prototype you build should just use window.prompt and window.alert for the user interface. Normally in production, you would avoid the use of these window functions, but we want to focus on functional programming without the convolution of DOM manipulation.

To assist you in the development process, your manager has set up scrum goals for you (seen below). You must complete the scrum goals in order.

As always, make sure you ask any questions you have before starting to code. In this case, take a look at the starter code. Does everything make sense?

Goal 1 — User Interface — Fork the repo and review the existing code. Your first goal is to build out the user interface (UI) using alerts and prompts.

The user should first be able to search for a person by name or by traits (see goal 5). Then they should be able to look up that person's information, the names of their descendants, and the names of their immediate family members.

The user interface you develop should be intuitive and not leave room for confusion. If a user enters something incorrectly, give them a message explaining what they may have done wrong, and prompt them again. Make sure you handle errors 'gracefully'.

Goal 2 —Get A Person's Info— Build out the functionality for displaying a person's information from their full name (first and last).

Goal 3 —Get Descendants— Build out the functionality for returning a person's descendants. The user just needs the list of their full names. Descendants are limited to relatives 'by blood'. *Must use recursion.*

Goal 4 —Get Family— Build out the functionality for returning a person's immediate family. The user just needs the list of their full names. Immediate family includes a person's parents, siblings, spouse (current only), and children. *Must use iteration.*

Goal 5 —Search and Filter— Build out the functionality for finding a person based on search criteria. The user may enter multiple (up to 5) characteristics of a user. The user should get a list of people back that meet all of the criteria. *Must use some combination of Array.Map( ) and Array.Filter( ).*

Types of terms you must accept:

- Age
- Height
- Weight
- Occupation
- Eye color

Github

https://github.com/devCodeCamp/starter-most-wanted

At the start of this project, fork this repo and rename it to 'most-wanted'. Before starting any development, read through the scrum goals and review the starter code. Notice how everything is organized.

Evaluation

This project is evaluated based on the completion of the goals. Use best practice, not only while coding, but with Git as well. Make sure that your code is refactored and pushed to Github by the deadline. Make sure to remove comments from your code, double check naming conventions, remove unnecessary files, and fix any errors. Each goal will receive a score based on functionality, coding best practice, and Git use.

Grading Rubric

Scrum Goal	Score	Points
Does the UI make sense?		5
Can you filter users by 1 criteria?		10
Can you filter by 2-5 criteria?		20
Can you look up a user's information?		15
Can you look up a user's descendants?		25
Can you look up a user's immediate family?		20
Did the developers make good commits?		5
Your final score is ____________ / 100		