📋 The Brief:

🥳 You are to create an application capable of parsing a CSV file and querying the results.

❗ You don’t have to spend a long time on this task. Aim to complete it using no more than 2-3 hours over the next few days.

👉 Read through **all of the instructions and criteria** before beginning.

💻 The Application

👉 Use C# to create a .NET application. This can be a standard Console-based application, a Web API, or even a WinForm.

However, **don't be overly ambitious or challenge yourself unnecessarily**. Choose whatever seems simplest and that you will find the most comfortable to achieve the desired outcome given your current knowledge!

👉 Your application should read from the provided *input.csv* file

This "comma-separated variables" file contains (fake!) data for 500 people:

- First name
- Last name
- Company
- Address
- City
- County
- Postal
- Phone1
- Phone2
- Email
- Web

👉 Load this data into a data structure suitable for querying.

💡There are packages available to parse CSV files into standard C# class-based Models. However, **it would be great practice to write your own simple parser**!

Make sure to use **separation of concerns** when implementing your parser. Your design should make it easy to swap your parser for another without changing any other code.

This article may be a useful starting place: Rolling Your Own? - A Simple CSV Parser Example.

👉 Your program should allow for the user to choose a specific output from a list (defined below). Depending on what kind of application you chose to create, the user input could be via choosing an option from a menu, or submitting a HTTP request to your application, or passing a parameter via the console, or changing a configuration file, or…

👉 Each program output consists of a **count** and then a **list of people** which correspond to some criteria. Each person should be displayed on a **new line** consisting of their **position in the list** and **their name** and **their company.**

For example, if the criteria was **“Every person who is called ‘Karma Quarto’”** then the output should be:

Count: 1

30 - Karma Quarto - J C S Machinery

👉 Here’s the list of criteria your program must be able to output according to the user requests:

- 🏠 OPTION 1: Every person who has “Esq” in their company name.
- 🏠 OPTION 2: Every person who lives in “Derbyshire”.
- 🏠 OPTION 3: Every person whose house number is exactly three digits.
- 🏠 OPTION 4: Every person whose website URL is longer than 35 characters (including the protocol and subdomain).
- 🏠 OPTION 5: Every person who lives in a postcode area with a single-digit value. (Note that in UK postcodes the “area” is the **first portion before the space**, so anything starting M3 or M7 would be acceptable, while anything starting M10+ would not. The portion after the space is not relevant.)

> 🏚 OPTION 6: Every person whose **first phone number** is **numerically larger** than their **second phone number**.

💥**OPTIONAL EXTENSION:** Once you're done, revisit your CSV parsing code. Can you add an option for the user to use either your custom parser, or one from a library?

Acceptance Criteria & Considerations

👉 Write ***production quality code***.

👉 Aim for your solution to be ***extensible***. Think about other features you might want to add in future, and design with them in mind.

👉 Ensure your code is ***well-tested***. This is best achieved using **Test Driven Development**.

👉 Your code must produce the correct output for various test inputs. We will run additional CSV files through your solution so make sure it behaves correctly for additional data too.

👉 Make sure you can justify and explain your design choices. Think about answers to questions like "Why did you create this class?" and "What other options did you consider?"

👉 Think about **flexibility** - what if your program had to load from a different CSV file, or combine multiple CSV files? How can you make the input and output flexible for future changes?

💥👉💥 IMPORTANT! Make sure your **git repo** is **clean**. Did you include a suitable *.gitignore*? Does the README look impressive when you first load the public repository page? Is your writing free of spelling errors? **Most importantly—ensure you CLONE your repo down to a fresh folder and try to run it.** Does it build and run without errors in a fresh folder?!

💡 Top Tips 💡

- Sketch / plan out your ideas first
- Commit into your Github repository frequently and with descriptive commit messages.

✅ How do I make a submission?

⭐ Submit your completed solution via a GitHub link by the deadline, on Google Classroom.

⭐ We will discuss your solution and extend it together during a mock interview.
