Google Sheets is a powerful spreadsheet application that allows users to organize, analyze, and visualize data. Apps Script is a cloud-based JavaScript platform that enables automation and integration with Google Sheets and other Google services.

To get started with Google Sheets and Apps Script, you can explore the following resources:

1 Google Sheets Overview

- Google Sheets is a web-based spreadsheet application that allows users to create, edit, and collaborate on spreadsheets in real time.
- It supports various functions for data manipulation, including formulas, charts, and pivot tables.
- Users can share sheets with others, allowing for collaborative editing and commenting.

Apps Script Overview

- Apps Script is a JavaScript-based platform that allows users to automate tasks across Google Workspace applications, including Google Sheets.
- It enables the creation of custom functions, menus, and user interfaces within Google Sheets.
- Users can integrate Google Sheets with other Google services, such as Gmail and Google Maps, to enhance functionality.

Getting Started with Apps Script in Google Sheets

1. Accessing Apps Script:
    - Open a Google Sheet.
    - Navigate to the menu: `Extensions > Apps Script`.
    - This opens the Apps Script editor where you can write your code.

With these steps, you can create a javascript code to make a custom function. Here are the custom functions we used.

1. ALAuthorsBioToDes.gs
    a. This function is used to extract education keywords from a biography,
    b. The first constant variable is keywords, such as “Education”, “Attended”, “lecturer”, “honors”, “taught”, “professor”, and “teacher” on line 2.
    c. The second constant variable is the list of alabamaUniversities, which is at line 3.
    d. On lines 24 and 26, RegExp constructor is used with two patterns.
        i. The first pattern is “keywordPattern”, which matches any of the keywords related to education followed by up to 5 words of any character.
        ii. The second pattern is “universityPattern” which matches any of the universities in Alabama followed by up to 5 words of any characters.
    e. The regular expressions are constructed as follows:
        i. `\\b` asserts a word boundary, ensuring that the keyword or university name is matched as a whole word.
        ii. `(" + educationKeywords.join("|") + ")` dynamically creates a group of keywords separated by the `|` (OR) operator.
        iii. `(\\s[A-Za-z]+){0,5}` matches 0 to 5 occurrences of a space followed by one or more alphabetic characters.
    f. Matching Against the Bio:
        i. The `bio.match(keywordPattern)` method is called to find all matches of the education keywords in the `bio` string. If no matches are found, it defaults to an empty array (`[]`).
        ii. Similarly, `bio.match(universityPattern)` is called to find matches for the university names.
    g. Combining Results:
        i. The results from both matches are combined into a single array using the spread operator (`...`) and `Set` to ensure that there are no duplicate entries.
    h. Reordering Results:
        i. The code checks if the string "education" is included in the results. If it is, it removes "education" from its current position and adds it to the front of the results array. This prioritizes "education" in the output.
    i. Returning the Final Result:

i. Finally, the results are joined into a single string with ": " as the separator and returned.

```
function extractEducationKeywords(bio) {
    const educationKeywords = ["education", "attended", "lecturer", "honors", "taught", "professor", "teacher"];
    const alabamaUniversities = [
        "Air University", "Alabama A&M University", "Alabama College of Osteopathic Medicine", "Alabama State University",
        "Amridge University", "Athens State University", "Auburn University", "Auburn University at Montgomery",
        "Bevill State Community College", "Birmingham School of Law", "Bishop State Community College", "Calhoun Community College",
        "Central Alabama Community College", "Chattahoochee Valley Community College", "Coastal Alabama Community College",
        "Columbia Southern University", "Community College of the Air Force", "Enterprise State Community College", "Faulkner University",
        "Gadsden State Community College", "H. Councill Trenholm State Community College", "Heritage Christian University", "Highlands College",
        "Huntingdon College", "Huntsville Bible College", "J.F. Drake State Community and Technical College", "J. F. Ingram State Technical College",
        "Jacksonville State University", "Jefferson State Community College", "Lawson State Community College", "Lurleen B. Wallace Community College",
        "Marion Military Institute", "Miles College", "Miles Law School", "Northeast Alabama Community College", "Northwest-Shoals Community College",
        "Oakwood University", "Reid State Technical College", "Samford University", "Selma University", "Shelton State Community College",
        "Snead State Community College", "Southern Union State Community College", "Spring Hill College", "Stillman College", "Talladega College",
        "Troy University", "Tuskegee University", "United States Sports Academy", "University of Alabama", "University of Alabama at Birmingham",
        "University of Alabama in Huntsville", "University of Mobile", "University of Montevallo", "University of North Alabama", "University of South Alabama",
        "University of West Alabama", "Wallace Community College", "Wallace Community College Selma", "Wallace State Community College Columbia College Missouri",
        "Edward Via College of Osteopathic Medicine", "Embry-Riddle Aeronautical University", "Florida Institute of Technology", "Southeastern University",
        "United States Army Command and General Staff College", "Alabama Presbyterian College", "Birmingham-Southern College", "Concordia College Alabama",
        "Daniel Payne College", "Judson College", "Southeastern Bible College", "Southern Benedictine College", "Virginia College"
    ];

    // Create a regular expression pattern that matches any of the education keywords followed by 0-5 words of any characters
    const keywordPattern = new RegExp("\\b(" + educationKeywords.join("|") + ")\\b(\\s[A-Za-z]+){0,5}", "gi");
    // Create a regular expression pattern that matches any of the Alabama universities followed by 0-5 words of any characters
    const universityPattern = new RegExp("\\b(" + alabamaUniversities.join("|") + ")\\b(\\s[A-Za-z]+){0,5}", "gi");

    const educationMatches = bio.match(keywordPattern) || [];
    const universityMatches = bio.match(universityPattern) || [];

    const results = [...new Set([...educationMatches, ...universityMatches])];

    if (results.includes("education")) {
        results.splice(results.indexOf("education"), 1);
        results.unshift("education");
    }

    return results.join(": ");
}
```

2. AlAuthorsDesToEd.gs
    a. On this file, there's a function named "extractEducationAlabama(cell)" to extract only Alabama Education from the previous function.
    b. This constant is used to identify the old universities and if the result is shown, it'll show as the new name or current name of the university. (Ex: Alabama Polytechnic Institute back in the 1900s are now Auburn University in current time

```
const oldNamesToNew = {
    "Alabama Polytechnic University": "Auburn University",
    "Alabama Polytechnic Institute": "Auburn University",
    "Alabama State Teachers College": "University of North Alabama",
    "Athens State Teachers College": "Athens State University",
    "Florence State Teachers College": "University of North Alabama",
    "Florence State College": "University of North Alabama",
    "Florence State University": "University of North Alabama",
    "Jacksonville State Teachers College": "Jacksonville State University",
    "Livingston State Teachers College": "University of West Alabama",
    "Livingston State College": "University of West Alabama",
    "Livingston State University": "University of West Alabama",
    "Montevallo State College": "University of Montevallo",
    "Montevallo State University": "University of Montevallo",
    "Troy State College": "Troy University",
    "Troy State University": "Troy University"
};
```

    c. Next, there's a constant citiesAndTowns which has a list of every single city and town, whether they are still active or not with the help of Wikipedia.

```
42    const citiesAndTowns = [
43        "Abbeville",
44        "Adamsville",
45        "Addison",
46        "Akron",
47        "Alabaster",
48        "Albertville",
49        "Alexander City",
50        "Aliceville",
51        "Allgood",
52        "Altoona",
53        "Andalusia",
54        "Anderson",
55        "Anniston",
56        "Arab",
57        "Ardmore",
58        "Argo",
59        "Ariton",
60        "Arley",
61        "Ashford",
62        "Ashland",
63        "Ashville",
64        "Athens",
65        "Atmore",
66        "Attalla",
67        "Auburn",
68        "Autaugaville",
69        "Avon",
70        "Babbie",
71        "Baileyton",
72        "Bakerhill",
```

d. Line 505 to 508 filters the 'alabamaUniversities' array to find any institutions that match the input 'cell'. It checks if the 'cell' includes the institution's name or an old name from 'oldNamesToNew' constant.
e. Line 511 checks if any of the found institutions are indeed part of the 'alabamaUniversities' list.
f. Line 514 checks if the 'cell' includes any of the cities or towns listed in 'citiesAndTowns'.
g. Lastly, 'line 517 to 524 are outputting the results. If any Alabama universities were found, it returns a comma-separated string of those institutions. If no universities were found but some cities were found, it returns a comma-separated string of those cities. If neither universities nor

cities were found, it returns a message indicating that no Alabama Institutions or cities were found.

3. ALAuthorsLatitude.gs and ALAuthorsLongitude.gs
    a. The names of the files are for functions with longitude and latitude from both Alabama Institutions and cities. With the help of Codenium (an extension from Visual Studio that's an AI help like Github Copilot), it managed to list every single coordinate. The only problem is that because there were so many cities, the AI needed to list the coordinates by separating some sections in alphabetical order. For example, cities from A to F were listed first, then H to M, and N to Z.

```
function getUniversityLatitude(universityName) {
    // Create an object to store universities and their updated latitudes
    const universities = {
      "Air University": 32.378259,
      "Alabama A&M University": 34.7838409,
      "Alabama College of Osteopathic Medicine": 31.2060765,
      "Alabama State University": 32.3629762,
      "Amridge University": 32.3626707,
      "Athens State University": 34.8059683,
      "Auburn University": 32.5980549,
      "Auburn University at Montgomery": 32.3695951,
      "Bevill State Community College": 33.836824,
      "Birmingham School of Law": 33.5116487,
      "Bishop State Community College": 30.695991,
      "Calhoun Community College": 34.647351,
      "Central Alabama Community College": 32.9244717,
      "Chattahoochee Valley Community College": 32.4244948,
      "Coastal Alabama Community College": 30.8531946,
      "Columbia Southern University": 30.3292725,
      "Community College of the Air Force": 32.4061022,
      "Enterprise State Community College": 31.2983627,
      "Faulkner University": 32.3848554,
```

```
80      //List of cities of Alabama universities
81      const cities = {
82          "Abbeville": 31.5718442,
83          "Adamsville": 33.5909379,
84          "Addison": 34.2012038,
85          "Akron": 32.8768007,
86          "Alabaster": 33.2442813,
87          "Albertville": 34.2670376,
88          "Alexander City": 32.9448429,
89          "Aliceville": 33.1290088,
90          "Allgood": 33.9031598,
91          "Altoona": 34.0273171,
92          "Andalusia": 31.3082308,
93          "Anderson": 34.9262032,
94          "Anniston": 33.6598291,
95          "Arab": 34.3181471,
96          "Ardmore": 34.9889765,
97          "Argo": 33.687605,
98          "Ariton": 31.5934976,
99          "Arley": 34.081491,
100         "Ashford": 31.184062,
```

```
545     // Check if the university name exists in the object and return the latitude
546     if (universities[universityName]) {
547       return universities[universityName];
548     }
549     // Check if the city name exists in the object and return the latitude
550     else if (Object.keys(cities).some(key => universityName.toLowerCase().includes(key.toLowerCase()))) {
551       return cities[Object.keys(cities).find(key => universityName.toLowerCase().includes(key.toLowerCase()))];
552     } else {
553       return "Not Applicable";
554     }
555   }
```

b. Lines 546 and 547 check if the ‘universityName’ exists as a key in the ‘universities’ object. If it does, it returns the corresponding latitude.
c. Lines 550 to 556 show two results:
    i. If the university name is not found, this line checks if any of the keys (city names) in the ‘cities’ object are included in the ‘universityName’ (case-insensitive).
    ii. If a match is found, it retrieves the corresponding latitude from the ‘cities’ object.
d. The other file shows the same results except it’s a longitude.

In conclusion, by using these four custom functions, it managed to extract every single educational information from the biography. While the parser would help, there may be a ton of information that a parser would need. The only downside for this use is that because every function contains so much data that needs to be extracted from the biography cell and other cells, it takes a long time to load the result. This resulted in a longer time whenever the Google Sheets is refreshed; therefore, once the results are showing fully, it need to be saved as a CSV file, not an ‘xlsx’ file because Microsoft Excel cannot contain custom functions and by saving as ‘CSV’ file, it’ll show the results only.