Maddox Scott

Ms. Sakthikumar

IB Computer Science, Period 5

20 March 2023

Bruhh

**Criterion C: Development**

**Key Techniques**

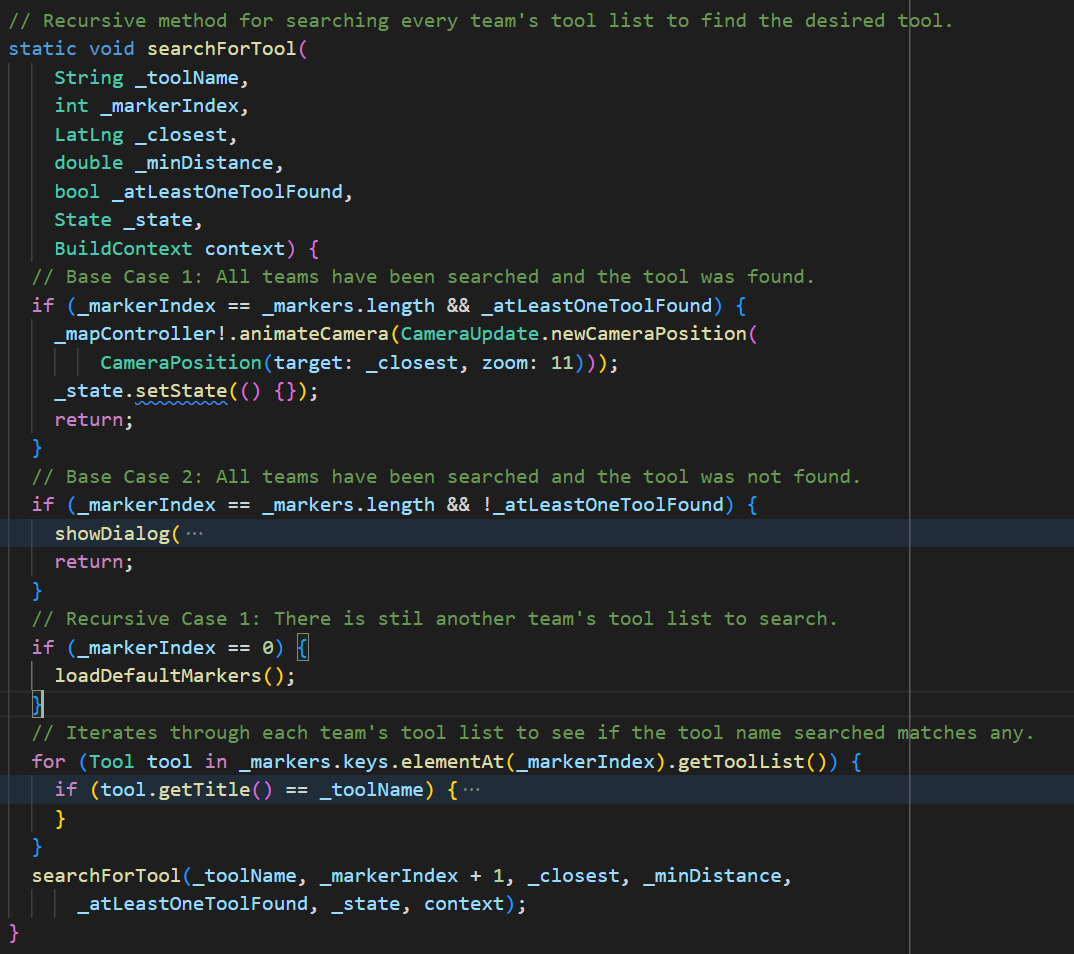
1. Recursively searching each team to find desired tools.
2. Writing to and parsing a nonvolatile txt file for local data storage.
3. Utilizing nested loops to iterate through a two-dimensional array.

**Minor Techniques**

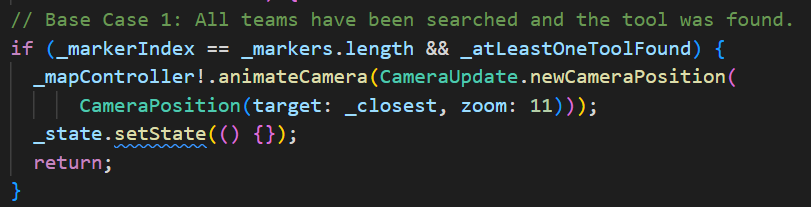
1. Use of the Google Maps and Google Places libraries.
2. User defined objects with unique variables, methods, and constructors.
3. Encapsulation of variables and methods within their respective classes.
4. Sorting a list of Strings by insertion.
5. Use of stack abstract data structure.
6. Use of set abstract data structure.
7. Use of the Model-View-ViewModel (MVVM) software design architecture to organize code.
8. Handling user input errors and exceptions.

**Key Technique Explanations**

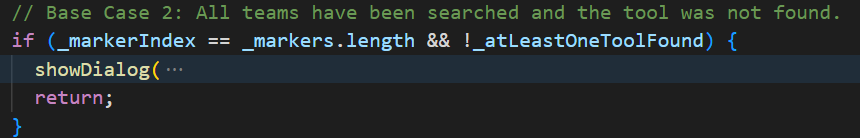
1. **Key Technique: Recursively searching each team to find a desired tool.**
   1. Explanation of Technique: The user is capable of entering in the name of a tool into a text field to search every team within the application to find the closest instance of that tool relative to the user’s location. A recursive method called “searchForTool()” is responsible for searching each Team instance to return the closest Team with the tool:



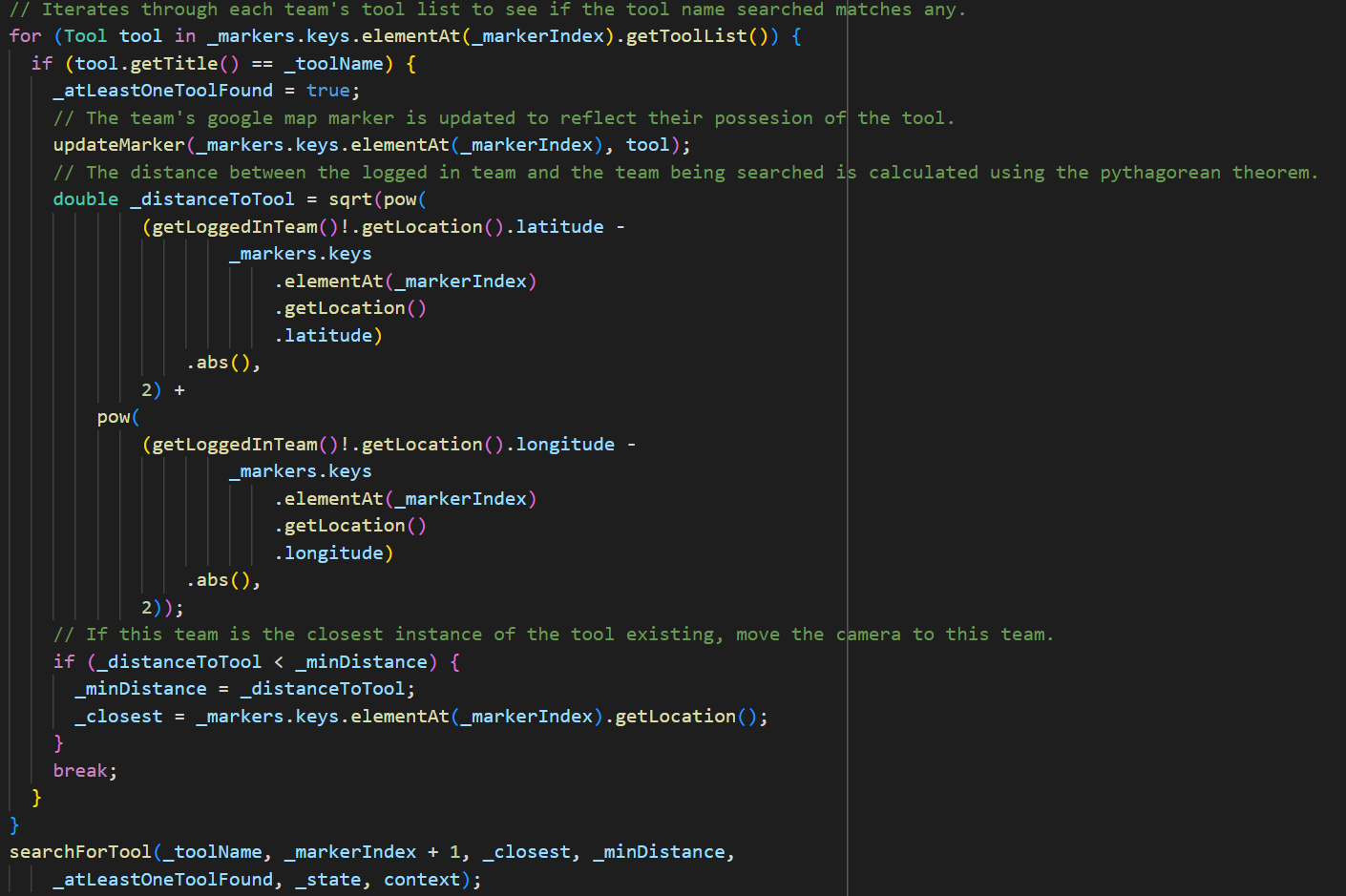
The method requires several parameters, such as the user input String, current index of the hashmap of Teams being searched, the closest tool’s location, and whether at least one instance of the tool has been found yet. To prevent the method from calling itself indefinitely, there are two base cases which prompt the termination of the recursive search:



First, if the index of the Team has exceeded the hashmap’s length and at least one existence of the tool has been found, the Google Map’s camera is moved to the team which possesses the tool and is closest to the user’s location.



If every team has been searched and the desired tool was not found, an error dialog is displayed, suggesting that the user submits an emergency request. When there are still more teams to be searched, the method proceeds with its recursive case:



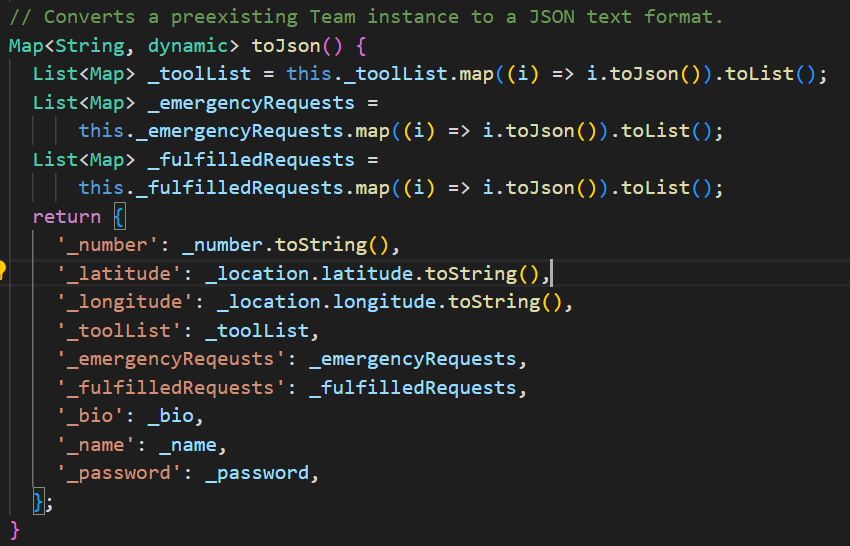
The current team’s toolList is iterated through, checking if tools’ names match the desired tool’s name. If so, “\_atLeastOneToolFound” is set to true. The distance between this team and the logged-in team is then calculated using the Pythagorean theorem. If the distance between these teams is less than the previously closest team in possession of the tool, this team’s location becomes the new “\_closest” LatLng variable for future comparison. Lastly, the recursive case calls the search method once more using the hashmap’s next index.

* 1. Ingenuity: By using recursion to search for the existence of the desired tool, a higher degree of freedom is allowed when determining what the application should do when the method terminates by utilizing several base cases. After every team has been searched, the method makes an informed decision based on if the tool has been found and what the currently closest team is. Recursion also allows for the code to be neatly organized into base and recursive cases, improving readability (Advantages 2).
  2. Appropriateness: As per success criteria 3, the application necessitates the ability to search for the nearest instance of a tool. This recursive method is scalable to any number of teams which may have been added to the Team hashmap. Also, for each recursive call, updated parameters are utilized to reflect the new closest team’s position and whether or not the tool has been found, as recursive methods can be called with different parameters each time (Introduction 3).
  3. Sources:

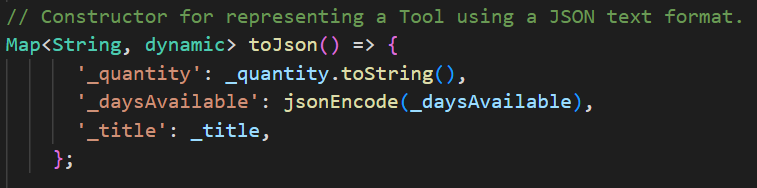
"Advantages and Disadvantages of Recursion." *Educative*, www.educative.io/courses/recursion-for-coding-interviews-in-cpp/qAx1lwQYDNG. Accessed 19 Mar. 2023.

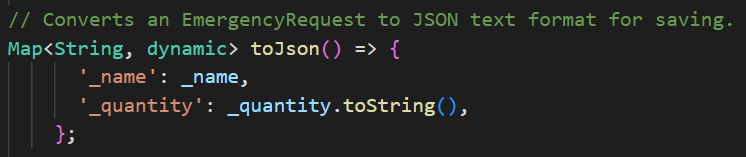
"Introduction to Recursion – Data Structure and Algorithm Tutorials." *Geeks for Geeks*, 17 Mar. 2022, www.geeksforgeeks.org/introduction-to-recursion-data-structure-and-algorithm-tutorials/. Accessed 19 Mar. 2023.

1. **Key Technique: Writing to and parsing a nonvolatile txt file for local data storage**.
   1. Explanation of Technique: In order to ensure that every team’s data persists between shutdown cycles, all Team instances are converted into a JSON text format and saved within a nonvolatile txt file. The Team class possesses a “toJson()” method which returns a map of strings, with each key correlating to an essential variable possessed by the team.

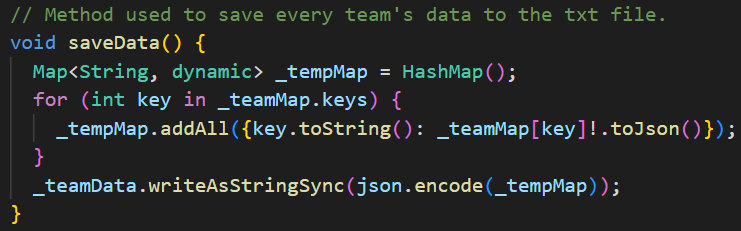


Several variables, such as the Team’s \_toolList and \_emergencyRequest arrayLists are composed of user-defined classes, which must also have their own toJson() methods. Thus, these lists are first converted to lists of string Maps too.

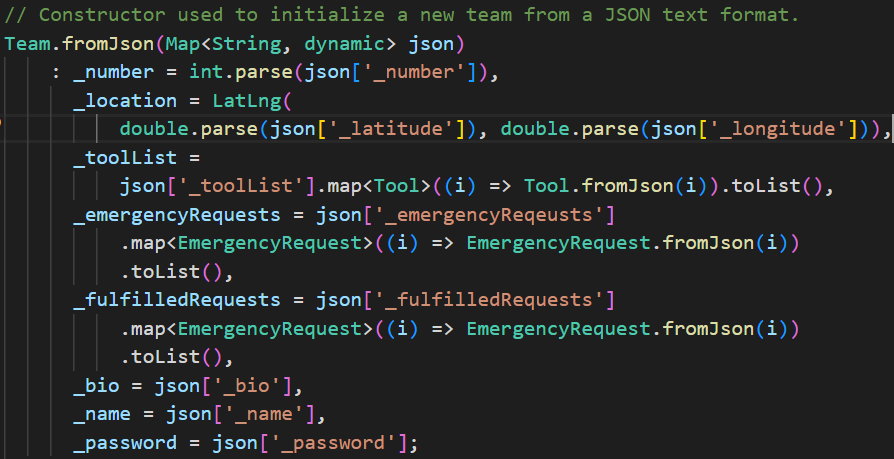




After every Team instance within the hashmap of Teams is converted, a new hashmap is created using each Team’s JSON Map and their identifying team number. Because every variable of every team is now in a string format, this temporary hashmap can be saved to the txt file.



Upon reloading the application, the txt file is parsed to restore the hashmap of each Team’s JSON strings. The Team class possesses a special constructor for creating a new instance from a JSON string, which is then used to repopulate the team hashmap.



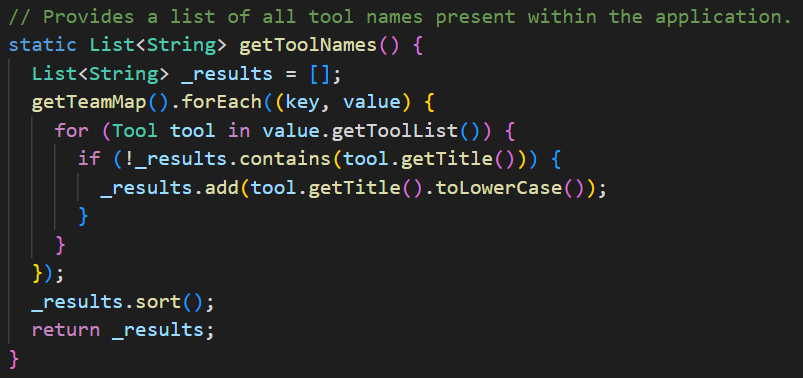
* 1. Ingenuity: Rather than save every Team’s JSON String in an unordered sequence, storing them first into a Map structure allows for every Team’s data to be retrieved in a Map structure, allowing for the instant repopulation of the “\_teamMap” variable upon application reload with no additional steps. Also, it allows for a single Team’s data to be retrieved without parsing the entirety of the txt file.
  2. Appropriateness: Every Team possesses a password which could be disastrous if its stored location was accessible to others. By saving these crucial values only to the local device in JSON format, it prevents outside influences from tampering with, stealing, or deleting these values. (JSON 12).
  3. Sources:

"JSON Defined." *Oracle*, www.oracle.com/database/what-is-json/#:~:text=readable%20JSON%20file.-,Why%20JSON%20is%20popular%20with%20developers,no%20additional%20code%20for%20parsing. Accessed 19 Mar. 2023.

1. **Key Technique: Utilizing nested loops to iterate through a two-dimensional array.**
   1. Explanation of Technique: On the “Search for Tools” page, the search field will display suggestions based on every Tool present within the application:



To retrieve the names of every tool within the application, the hashmap of Teams is iterated through. For each Team, the entirety of the Team’s toolList is iterated through. For every element of the toolList, the Tool’s title is added to a list of Strings if it has not yet been added. Lastly, the list of tool names is sorted and returned to the Autofill class to be displayed as suggestions.



* 1. Ingenuity: Although the application would function the same if duplicate tool names appeared as suggestions, excessive clutter harms the user experience and ergonomics of the application (Minhas 8). By avoiding duplicates and displaying suggestions alphabetically, the application’s readability is improved.
  2. Appropriateness: Utilizing autofill suggestions enables the application to account for deviations in tools which should share similar names. If many teams possess tools named “duct tape” but the user unsuccessfully searches for “a roll of tape,” the user might assume that no team possesses tape, unless tools with similar names are suggested.
  3. Source:

Minhas, Saadia. "The Eight Rules of Mobile Design." *Medium*, 10 Aug. 2019, uxdesign.cc/8-rules-of-mobile-design-1b8d9936c241. Accessed 19 Mar. 2023.

**Word Count: 906**