Administrative:

Team Name: "CRTL+EAT+DEL"

Team Members: Leila Ossi

GitHub URL: https://github.com/leilaossi/Project-3

Link to Video: https://youtu.be/pRvmXLqxcd4

Extended and Refined Proposal:

1. Problem: What problem are we trying to solve?

   The problem I aim to address is the lack of easy access to comprehensive nutritional information about various food items. Many individuals, whether due to dietary preferences, health concerns, or general well-being, seek information about the nutritional content of the food they consume. However, obtaining accurate and readily available data can be challenging.

2. Motivation: Why is this a problem?

   The problem is significant for several reasons:
   - Health Awareness: In today's health-conscious society, people are increasingly concerned about the nutritional value of the food they consume. They want to make informed choices that align with their health goals and dietary requirements.

   - Diverse Dietary Needs: People have diverse dietary needs and restrictions. Some may be looking for low-calorie options, while others focus on protein intake or need to manage specific nutrients due to medical conditions. Access to detailed nutritional data helps individuals tailor their diets to meet their unique requirements.

   - Weight Management: For those aiming to manage their weight, understanding the caloric content and macronutrient composition of foods is crucial. Access to accurate nutritional information assists in tracking daily caloric intake.

   - Education and Awareness: Providing easy access to nutritional data fosters education and awareness about healthy eating habits. It empowers individuals to make informed choices about their diets, potentially reducing the risk of diet-related health issues.

3. Features implemented:

   I have implemented features that allow users to input a food item and view its nutrition facts, as well as providing search options using Ternary Search and Jump Search algorithms. When either search is used, the program will also output the time it took the search to complete. I also offer a menu with options to view food titles, food categories, and exit the program.

4. Description of data:

   ABBREV_alphaIndexed.csv (8790 rows * 53 columns): This csv file (the dataset) contains the nutritional information for various food items.
   The dataset originally was not alphabetized and the original NDB_No beside the shrt_Desc column was changed to the food items index number. I changed both for easier searching purposes.
   Also, the website I linked in the references beside “Dataset:” will also include more specific information, where all the data is explained in an attached PDF document.

5. Tools/Languages/APIs/Libraries used:

   As a tool I am using the CLion IDE.
   The language is strictly C++ for the development of the program.
   I am using these C++ libraries: ‘iostream’, ‘vector’, ‘string’, ‘fstream’, ‘sstream’, ‘array’, and the standard for ‘std’ and ‘chrono’.
   Lastly, I use no APIs.

6. Algorithms implemented:

   I have implemented Ternary Search and Jump Search algorithms for searching food items in the dataset.
   The Ternary Search algorithm is implemented in the ‘ternarySearch’ function, which performs a ternary search on a sorted array to find a target value. It uses recursive logic to divide the array into three parts.
   The Jump Search algorithm is implemented in the ‘jumpSearch’ function, which performs a jump search on a sorted array. It calculates a jump step size and then searches for the target value by making jumps within the array.
   I compare them based on how long they each take to search for the correct food item.

7. Additional Data Structures/Algorithms used:

   - Struct ‘FoodRecord’: I defined a ‘FoodRecord’ struct that stores information about food items. While not a traditional data structure, it can be considered a user-defined data structure for organizing and storing food data.

   - Vector ‘names’: Within the ‘FoodRecord’ struct, I have a vector of stored strings called ‘names’. It is used to store the names of food items extracted from the CSV.

   - ‘displayCSV’ function: While the function itself is not a data structure, it interacts with and utilizes data structures. It creates and operates on instances of the ‘FoodRecord’ data structure.

- 'getNames' function: This function returns a vector of strings that contain a combination of 'NDB_No' and 'shrt_Desc' from the CSV data, representing food names. This function uses a vector to store and return these names.

8. Distribution of Responsibility and Roles: Who did what?

   Throughout the project, I worked alone. So, I wrote all the code by myself, such as the user interface, functions, searching algorithms, etc.

Analysis:

1. Any changes the group made after the proposal? The rationale behind the changes?

   At first, I wanted to compare two data structures: a Hash Table against a Trie function. However, I soon discovered they are not comparable data structures that achieve the same task. Therefore, I decided to compare two searching algorithms: a Ternary Search against a Jump Search.
   I found the algorithm approach to be more attainable to complete considering that I was now working individually in a short amount of time. I also thought these two would work very well when being implemented with my dataset and user interface.
   Then I decided that instead of having the food items assigned to a specific category before outputting the nutritional information, I wanted to assign each food title to an indexed number and having the user then input that number rather than the food title.

2. Big O worst case time complexity analysis of the major functions/features you implemented:

   - 'ternarySearch': The worst-case time complexity is O(log3(n)), where n is the number of elements in the 'numbers' vector. Ternary search divides the search range into three parts in each iteration, reducing the search space significantly. This results in a time complexity of O(log3(n)) since the search space is divided by three in each step.

   - 'jumpSearch': The worst-case time complexity is O(sqrt(n)), where n is the number of elements in the 'numbers' vector. Jump search involves making jumps within the array, with the jump step size determined by the square root of the array size. In the worst case, the algorithm may require 'sqrt(n)' jumps to find the target element.

   - 'getNames': The worst-case time complexity is O(n), where n is the number of lines in the CSV file. In the function, it iterates through each line of the CSV file to extract and construct food names. This operation has a linear time complexity as it processes each line once.

   - 'displayCSV': The worst-case time complexity is O(1). The function performs a fixed number of operations to display nutritional for a specific food item. It does not depend on the size of the dataset or the number of elements in the vector. Therefore, it has constant time complexity.

- Menu Interface and User Input Handling: The worst-case time complexity is O(1). Handling the menu interface and user input involves basic input/output operations and conditional statements. These operations do not depend on the size of the dataset and have constant time complexity.

- CSV Data Parsing: The worst-case time complexity is O(n), where n is the number of lines in the CSV file. The process of reading and parsing data from the CSV file involves iterating through each line and parsing the data fields. As with the 'getNames' function, this operation has a linear time complexity.

Reflection:

1. As a group, how was the overall experience for the project?

   For me, I had a group the beginning, but not soon after the proposal that I completed and submitted by myself, they both became unresponsive, so a couple days before the deadline I decided I wanted to work individually because I was not comfortable sharing any credit with them as they never contributed anything. So, I would say my overall experience was quite poor and anxiety-inducing because I had to complete the project in such a short amount of time. Nevertheless, I am still very proud of the work I did accomplish despite the circumstance and time constraints.

2. Did you have any challenges? If so, describe:

   I experienced several challenges throughout this project. First, was handling input validation. Ensuring that user inputs are valid, within the expected range, and of the correct data type is essential for the program's readability. I used while loops and conditional statements to repeatedly prompt users for valid input until it met the specified criteria. I also used functions like 'cin.clear()' and 'cin.ignore()' to clear the input stream and handle invalid input efficiently.
   Second, was parsing the CSV data accurately. I needed to extract specific fields (e.g., integers, doubles, strings) from the CSV file and store them correctly in data structures. I utilized the 'std::ifstream' and 'std::getline' functions to read lines from the CSV file. I also used string manipulation and type conversion functions ('atoi', 'atof') to extract and convert data from the CSV file.
   Third, was optimizing the search algorithms for performance as I was dealing with a large dataset. I carefully designed and tested the search algorithms to ensure they correctly locate food items within the dataset. I also measured and reported the elapsed time for the search operations to assess their performance.

3. If you were to start once again as a group, any changes you would make to the project and/or workflow?

If I were to work in a team, I would first make sure that I really know the people in my group and know their style of communicating and their efficiency with completing assignments on time. Then, I would consider having a more structured workflow with more defined roles and responsibilities. Additionally, I would consider expanding the project by adding more features or improving the user interface.

4. Comment on what each of the members learned through this process:

   Through this project, I gained better experience in C++ programming, working with CSV files, implementing search algorithms, and creating a console-based user interface. These skills can be valuable for future projects and programming tasks.

References:

1. My dataset used: https://data.world/awram/food-nutritional-values

2. Parsing the CSV: https://gist.github.com/fernandozamoraj/aa35555a56884242041495cbb654dbe8

3. Ternary Search: https://www.geeksforgeeks.org/ternary-search/

4. Jump Search: https://www.geeksforgeeks.org/jump-search/