WORD DICTIONARY
IMPLEMENTATION USING **BINARY SEARCH TREE**

Course: CSE207

Section: 09

Group - 05

- Junnun Mohamed Karim (2022-1-60-108)
- Md Murad Khan Limon (2022-1-60-044)
- Md. Yousuf Hozaifa (2022-1-60-162)

PROJECT OBJECTIVE

- **Build a Functional Word Dictionary:**

  The primary goal of this project is to create a word dictionary that functions as a reference book, providing alphabetical listings of terms with their meanings and applications.

- **Implement Key Functionalities:**

  Implement essential functionalities efficiently, including the ability to seamlessly add new words, delete existing words, update word entries, and perform word searches within the dictionary.

- **Utilize Binary Search Tree (BST):**

  Employ a Binary Search Tree data structure to efficiently store and manage a list of 50 words in alphabetical order.

- **File Data Input:**

  Enable the dictionary to populate its initial data from a text file, with each entry containing a word and its corresponding definition (meaning).

FEATURE

- **Core Functionalities:**
  - ***Add Word***: Easily expand the dictionary with new words and definitions.
  - ***Search Word***: Quickly find word meanings and applications.
  - ***Delete Word***: Efficiently remove unnecessary entries.
  - ***Update Word***: Modify word entries with new values.
  - ***Display Word-Tree***: Visualizes the entire contents of the Binay Search Tree.
- **Enhanced User Experience:**
  - ***Suggested Words***: Offers suggestions for similar words during searches.
  - ***Error Handling***: Provides clear messages for invalid input and errors.
- **Version Control and Build System:**
  - ***Git***: Utilized for version control and collaborative development.
  - ***CMake***: Implemented for automated and platform-independent building.

DATA STRUCTURE

- **Binary Search Tree (BST):**
  - Purpose: Efficiently stores and organizes dictionary entries.
  - Characteristics: Nodes hold word entries; maintains alphabetical order.
  - Advantages: Quick word retrieval with logarithmic complexity.
  - Usage: Main dictionary storage.

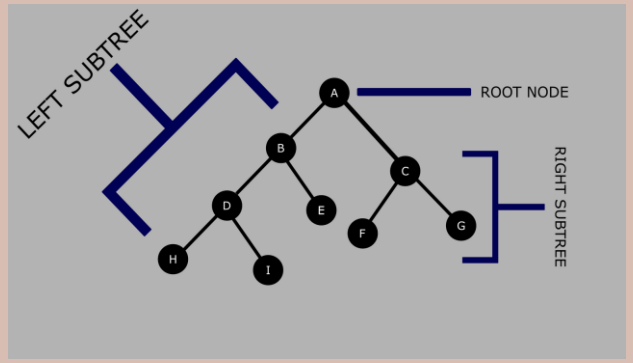


Figure: Binary Search Tree diagram

- **Linked-List:**
  - Purpose: Generates word suggestions during searches.
  - Characteristics: Sequential nodes store suggested words.
  - Advantages: Dynamically stores and supports easy insertion/removal.
  - Usage: Displaying search suggestions, loading entries from file.

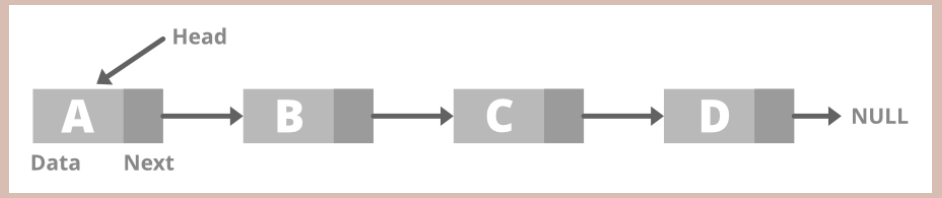
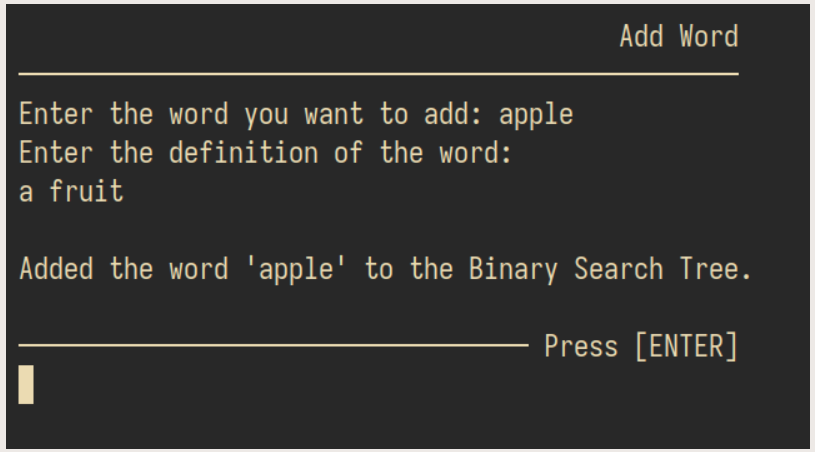


Figure: Linked-List diagram

PROGRAM FLOW

- Initialization and Data Loading
- Main Menu Loop
- User Choice Selection
- Option Execution
- Repeat or Quit
- Exit and Cleanup

MAIN MENU LOOP:
ADD WORD



User Input:
- Prompt user to enter a word for addition.
- Validate input and ensure it doesn't already exist in the dictionary.

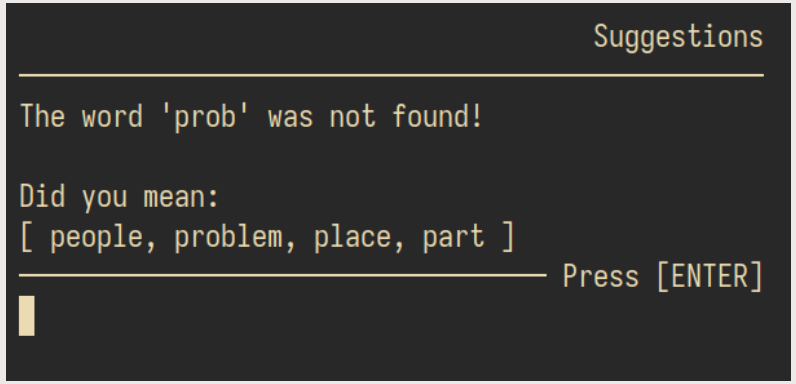
Input Definition:
- Request user to input the definition (meaning) of the word.

Word Object Creation:
- Create a word object with the provided word and definition.

Add to BST:
- Check if the word exists in the Binary Search Tree (BST).
- If not, insert the new word into the BST.

Feedback:
- Provide a success or error message based on the operation's outcome.

MAIN MENU LOOP:
SEARCH WORD



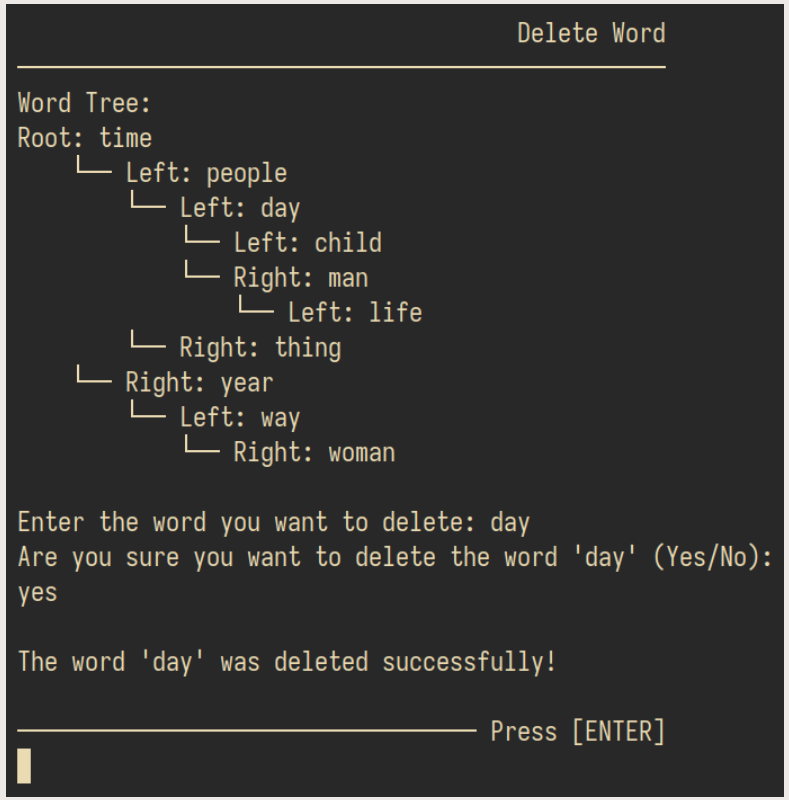
User Input:
- Prompt user to enter a word for searching.
- Validate input.

BST Search:
- Search the Binary Search Tree (BST) for the entered word.
- If found, display the word's definition.

Not Found Handling:
- If the word is not found, call print_suggestions() to provide suggestions based on a similar substring.

Print Suggestions Function Flow:
- Closest Node Search:
- Determine the closest node in the BST based on the input word.
- Suggestion Generation:
  - Recursively search and populate a list of word suggestions that start with a matching substring.
- Display Suggestions:
  - Display the suggestions to the user, enhancing the search experience.
- User Interaction:
  - Wait for user input or further actions after displaying suggestions.

MAIN MENU LOOP:
DELETE WORD



User Input:
- Prompt the user to enter the word they want to delete from the dictionary.
- Validate user input to ensure it's a valid word.

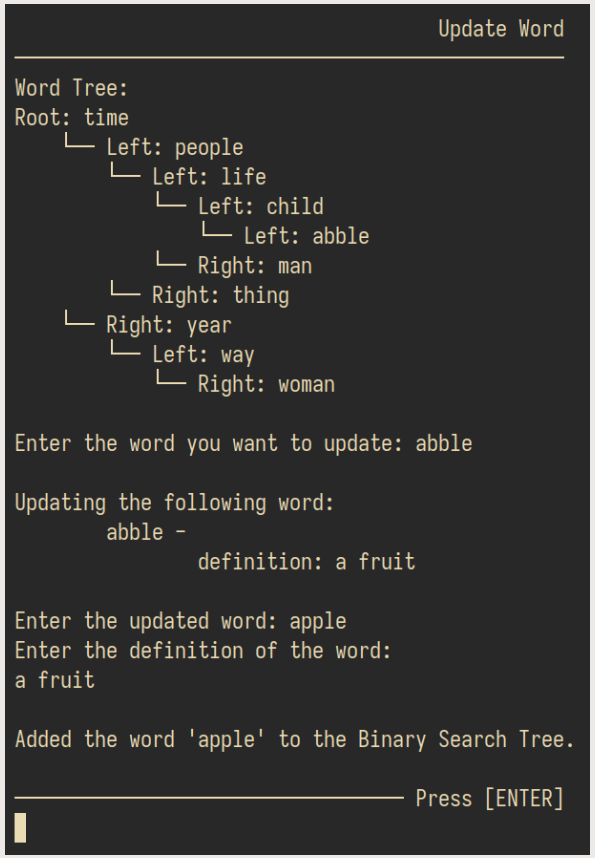
Word Existence Check:
- Check if the entered word exists in the Binary Search Tree (BST) by searching for it.

Confirmation:
- If the word is found in the dictionary, ask the user for confirmation to proceed with the deletion to avoid accidental removal.

Deletion from BST:
- If the user confirms the deletion, remove the word and its definition from the BST.
- Ensure that the BST structure is properly adjusted to maintain its properties.

Feedback:
- Provide a feedback message to the user, indicating whether the word was successfully deleted or if there was an error during the deletion process.

MAIN MENU LOOP:
UPDATE WORD



User Input - Target Word:
- Prompt the user to enter the word they want to update within the dictionary.
- Validate input to ensure it's a valid word.

Word Existence Check:
- Check if the entered word exists in the Binary Search Tree (BST) by searching for it.

Confirmation:
- If the word is found in the dictionary, ask the user for confirmation to proceed with the update.

Deletion from BST:
- If the user confirms the update, remove the existing word and its definition from the BST.

User Input - New Word and Definition:
- Prompt the user to input the updated word and its new definition (meaning).

Word Object Creation:
- Create a new word object with the updated word and definition.

Add to BST:
- Insert the updated word into the BST, effectively updating the dictionary with the new information.

Feedback:
- Provide a feedback message to the user, indicating whether the word was successfully updated or if there was an error during the update process.

CORE FEATURE

The two main features of our project are:

- The implementation of **word representation** within the Binary Search Tree.
- The incorporation of a **search suggestion** feature.

REPRESENTATION OF WORDS

A special "word" class is implemented to represent words in a way that makes them easy to manage and search. This "word" class has two important parts:

- **Term:** This is where we store the actual word, like "apple" or "computer." Think of it as the word itself.
- **Definition:** Here, we keep the meaning or definition of the word, like what "apple" or "computer" actually means.

To elaborate:

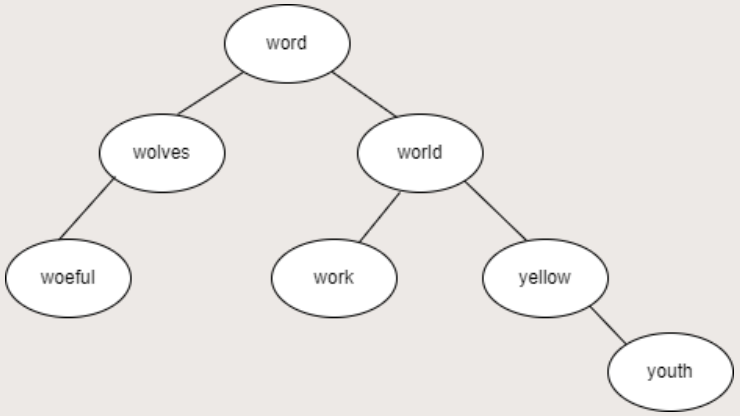
- Words are stored in a special container.
- The container holds both the word and its meaning.
- This method efficiently organizes words.
- Facilitates quick and easy word retrieval.
- Think of it like a well-organized bookshelf.

WORD SUGGESTION

- **User Input**: User enters a word to search for in the dictionary.
- **Search Attempt**: The system initially attempts to find the exact word in the Binary Search Tree (BST).
- **No Exact Match Found**: If no exact match is found in the BST, it means the word doesn't exist in the dictionary.
- **Partial Match Handling**: The system then analyzes the user's input and performs a search to find the closest matching prefix (a part of the user's input) using the **find_closest_node** function.
- **Generating Suggestions**: The **find_closest_node** function identifies the node with the closest matching prefix, which represents the closest word to the user's input.
- **Suggestion List Generation**: The **find_closest_node** function returns this closest node to the **print_suggestions_helper** function, along with the user's input.
- **Populating Suggestion List**: The **print_suggestions_helper** function recursively traverses the BST, looking for words that start with the same characters as the closest matching prefix.
- **Suggested Words List**: It populates a list of suggested words based on this search.
- **Display Suggestions**: Finally, the **print_suggestions** function displays a list of suggested words related to the user's input, which the user can consider for further exploration.

WORD SUGGESTION

Let’s think of an example, suppose the user searches for the word “yallow” in the following BST -



The find_closest_node function would return the node “yellow”. After that the print_suggestions_helper would print the word in the following subtree -

LIMITATION

Limited Suggestion Mechanism

• The suggestion mechanism for similar words is based on finding words with the same prefix. While this can be helpful, it might not provide accurate suggestions for words that are not found due to small typos or variations.

Performance Impact of String Manipulation

• The implementation heavily relies on string manipulation, such as transforming words to lowercase and extracting substrings. String manipulation operations can impact performance, especially when dealing with a large number of words and extensive searches.

Command-Line Interface Only

• The word dictionary's interaction is limited to a command-line interface (CLI). This might not provide the most user-friendly experience, especially for users who are not familiar with using CLI applications.