Project Title: Simple Text Editor with Undo/ Redo Functionality

1. Overview

This project implements a basic text editor that supports two primary operations—**insertion** and **deletion**—while maintaining a history of operations so that users can undo or redo their actions. Instead of using built-in stack data structures, both implementations (in Java and Python) use a custom linked list to store the operation history. This custom history structure provides last-in–first-out behavior by allowing new operations to be added and removed from the head of the list.

2. Components and Architecture

2.1 Operation Record (Command)

- **Purpose:**
  The operation record (or command) encapsulates all details of an edit operation so that it can be reversed (undone) or reapplied (redone).
- **Attributes:**
  - **operation:** A string indicating the type of operation. Allowed values are `"insert"` and `"delete"`.
  - **index:** An integer representing the position in the text buffer where the operation occurred.
  - **text:** The text that was either inserted or deleted.
- **Usage:**
  Every time the user makes an edit (insert or delete), a new command is created and stored in the operation history.

2.2 Custom Linked List for Operation History

- **Purpose:**
  Instead of using a built-in stack, the undo and redo histories are maintained using a custom linked list. This linked list supports:
  - **Adding an Operation:** New commands are added at the beginning (head) of the list.

- **Removing the Last Operation:** The most recent operation is removed from the head, providing last-in–first-out behavior.
- **Clearing the History:** All nodes can be removed from the list when needed.
- **Checking Emptiness:** To determine if there are any operations left to undo or redo.

- **Implementation Details:**
  - In **Python**, a `Node` class is used along with an `OperationHistory` class that manages the linked list.
  - In **Java**, a similar approach is taken using a `HistoryNode` class and an `OperationHistory` class.

2.3 Text Editor Class

- **Purpose:**
  The core class of the project, responsible for managing the text buffer and coordinating all operations.
- **Attributes:**
  - **Text Buffer:**
    - In Python: A string that holds the current text.
    - In Java: A `StringBuilder` that allows mutable string operations.
  - **Undo History:**
    A custom linked list that holds all performed operations (each new operation is added to the head).
  - **Redo History:**
    A custom linked list that holds operations that have been undone. When a new operation is performed, this history is cleared.
- **Methods:**
  - **insert(index, newText):**
    - **Description:** Inserts a given string (`newText`) at the specified position (`index`).
    - **Behavior:**
      - Validates that the index is within the bounds of the current text.
      - Updates the text buffer by inserting the new text.
      - Creates and adds a corresponding operation record to the undo history.
      - Clears the redo history because a new operation invalidates previous redo

options.

- **delete(index, length):**
  - **Description:** Deletes a substring of the given length starting from the specified index.
  - **Behavior:**
    - Validates the index and length (ensuring that the deletion range is valid).
    - Stores the text that is removed.
    - Updates the text buffer by deleting the specified characters.
    - Creates and adds a corresponding operation record to the undo history.
    - Clears the redo history.
- **undo():**
  - **Description:** Reverses the most recent operation.
  - **Behavior:**
    - Removes the most recent command from the undo history (using the custom linked list).
    - If the operation was an **insert**, its undo action is to remove the inserted text.
    - If the operation was a **delete**, its undo action is to reinsert the deleted text.
    - The reversed command is then added to the redo history.
- **redo():**
  - **Description:** Reapplies the most recently undone operation.
  - **Behavior:**
    - Removes the most recent command from the redo history.
    - Re-executes the operation (inserting or deleting the text as originally done).
    - The command is added back to the undo history.
- **get_text():**
  - **Description:** Returns the current state of the text buffer.
  - **Usage:**
    - This method is used within test cases to verify that the editor's state matches expected outcomes after a series of operations.