8. LAB H – COMMAND

8.1. Purpose

We wish to implement a small fraction of a word-processor. Our mini-editor will be able to perform commands and then undo them. We will implement the supplied `interface EditorCommand` and create 2 concrete command classes:

1. `class AppendCommand -` adds text.
2. `class DeleteCommand -` deletes text

Make sure you implement the undo() method.

setState

Lab Scenario (WHAT to do)	Examine the supplied code : `interface EditorCommand` will serve as a base for editor commands SimpleEditorToComplete will be the editor Develop the following EditorCommand subclasses: AppendCommand DeleteCommand And `SimpleEditor` (which contains the inner command classes)
Implementation Steps (HOW to do it)	**Code the `SimpleEditor` class** The editor class will allows clients to execute and undo commands which are implemented as inner classes Implement the following member variables: 1. list for storing commands 2. index pointing to the current command in the list 3. StringBuffer holding all the text Implement the addCommand method: 1. add the command at the right position using the index value 2. update the position 3. execute the command Implement the undo method: 1. decrease the index by 1 2. fetch the command from the list based on the index 3. undo the command Implement the redo method: 1. get the current command based on current index value 2. execute the command 3. increase the index by 1 Implement the AppendCommand inner class: 1. extend EditorCommand 2. add a String member variable holding the text to append 3. create the constructor taking one param reflecting the text to append 4. implement execute adding the text to the editor's buffer. As this is an inner class it has access to the outer class member variables 5. implement the `void undo()` method – decrease the editor size by the appended text size effectively

	deleting the text previously appended Implement the DeleteCommand 1. extend EditorCommand 2. define member variables a. int start b. int end c. String text (to delete) so we can undo later 3. implement the constructor taking int from, int to values as parameters. These values indicate what to delete 4. implement the execute method a. get the text to delete from the editor buffer variable (i.e. use substring) b. store it (for undo()) c. delete it from the text buffer 5. implement the `void undo()` method insert the deleted text to the editor's buffer from position start (using `buffer.insert(start, text)`)

8.2. Review of Command

Two classes exist and should be extended / modified:

1. EditorCommand an interface serving as base for concrete commands.
2. `SimpleEditor` as the base structure of the basic editor.

Two classes to create:

1. AppendCommand implemented as an inner class within SimpleEditor.
2. DeleteCommand implemented as an inner class within SimpleEditor.

The reason we use inner classes is is to gain access to the editor text buffer.

Note: The solution for this exercise is available in the 'solutions' directorysetState