Note: Similarity to this year's work or work from previous years results in automatic zeroing to the lesson of all involved.

Task 1 - Compressed Trie

Submission Deadline: Wednesday, March 30, 11:59 p.m

Write a program in java that implements a dictionary of the English language through a compressed Trie based (radix) on the English alphabet (az). Compressed is any Trie whose nodes that are not terminals and have only one child are joined to their child. Joined nodes represent a string instead of a single character. Supervisory example of how a compressed Trie is configuredcan be found here .

Your program should only contain the class files a) TrieNode.java which describes the node, Trie.java which describes the Trie structure and c) HW1.java which contains the main method. The above classes must be in the ce326.hw1 package.

Description of the operation of the program

The looping program prints the string " " followed by a space ?and a newline character, then reads from the keyboard and executes one of the following commands by printing to stdout.

		Printouts	
Mandate	**Description**	**Success**	**Failure**
- i word	Inserts word into Trie. Returns success if the word does not exist in the Trie and is successfully inserted into it. Otherwise it returns failure.	**ADD word OK***	**ADD word NOK***
- r word	Deletes the word word from the Trie. Returns success if the word exists in the Trie and is successfully deleted. Otherwise it returns failure.	**RMV word OK***	**RMV word NOK***
- f word	Searches for word in Trie.	**FND word OK***	**FND word NOK***
- p	Prints the pre-order penetration of Trie. Initially, the string "PreOrder: " and then follow the individual strings of the tree nodes printed during the preorder traversal. If a node is terminal the string "#" is printed immediately after its content (no space). After each string a blank character is printed and after the last blank character is printed one character is printed two newline characters are printed. **Note:**Its rootA trie that has no content is not printed		
- d	Prints all the words of the stored dictionary in alphabetical order. A newline character is printed first, the string "*****Dictionary *****" followed by newline character and then the words contained in the dictionary. A newline character is printed after each word, and two newline characters are printed after the last word instead of one.		
- w word X	Searches the Trie for all wordssame length with the word**word**which are farexactlyX characters from the given word. For example, ● the words boy and toy are one character apart. ● the words small and smell are one character apart. ● the words small and smile are two characters apart. ● the words newspaper and newsgroup are five characters apart.	A newline character is printed and the string " Distant words of W (X):"followed by character line break, where W is the word and X is the distance of the searched words. Then the words that are X characters away from the given word are printed in alphabetical order. A newline character is printed after each word, and two newline characters are printed after the last word instead of one.	

Note: Similarity to this year's work or work from previous years results in automatic zeroing to the lesson of all involved.

-s suffix	Searches the Trie for words containing the given suffix.	A newline character is printed and the string " Words with suffix S:" followed by a newline character, where S is the given suffix. Then the words containing the given ending are printed in alphabetical order. A newline character is printed after each word, and two newline characters are printed after the last word instead of one.
- q	Prints the string**"Bye bye!"** followed by a newline character and terminates the program.	

*A newline character is printed after each message.

Limitations - Remarks

- All words entered or searched for must contain only letters from the English alphabet (no punctuation marks, numbers or letters of any other alphabet). If a word does not obey the above restriction it is ignored and a failure message is returned the code ~~continues in the next iteration.~~

- Words or node contents are always printed in lowercase letters. How you store them in the structure (lowercase or uppercase) is up to you.
- Words to enter can contain both upper and lower case letters (eg sMaLl, SMALL or small).

Reading from the keyboard

The following code uses the classjava.util.Scanner to repeatedly read from the keyboard. Do not copy-paste the following code, because characters are inserted that do not is ASCII . Word-by-word reading is more convenient for your program.

Read word for word	Read line by line

import java.util.Scanner;
 	
public class ReadWord {
 	
public static void main(String []args) {
Scanner sc = new Scanner(System.in); String word?
while(sc.hasNext()) {
word = sc.next();
}
}
}
	import java.util.Scanner;
	
	public class ReadLine {
	
	public static void main(String []args) {
	Scanner sc = new Scanner(System.in); String line?
	while(sc.hasNextLine()) {
	line = sc.nextLine();
	}
	}
	}

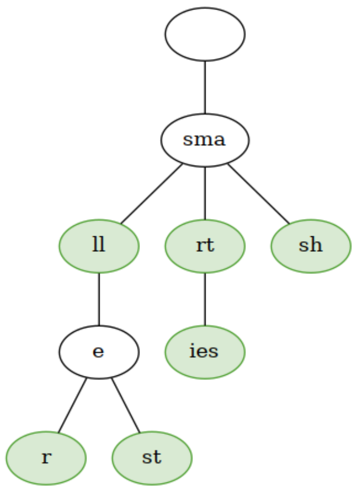
Print Trie to Image (for debugging only)

Printing the Trie to an image is not requested by the task. But you can implement it to make your code easier to debug.

To print a graph in an image you can use the programdot of the suite graphviz . To do this it is enough to create a text file (with a .dot ending in its name) that describes the structure based on the requirements of the above program. The print series of

Note: Similarity to this year's work or work from previous years results in automatic zeroing to the lesson of all involved.

of nodes of the tree is its pre-order traversal. The appearance of an edge may precede the appearance of the tree node within the file.



For example, for the adjacent Trie with records small, smaller, smallest, smart, smarties andsmash you can downloadthe indicative dot file here .

In the above file, each node has a unique identifier consisting of an integer. In the file, the integer related to the node with "sma" contents is1418481495.

The unique integer for each node results from the function's return valuehashCode(). The function hashCode() is present in all classes.

After writing the dot file you can create a PNG image provided you have the graphviz suite installed on your computer. The transformation command is as follows:

$> dot -Tpng input_filename.dot -o output_filename.png

Wherever it is**input_filename.dot** the name of the dot file you made earlier and **output_filename.png**is the name of the resulting final image.

Print string to file

The following program prints the string fileContents in the file named fileName which if it does not exist creates it and if it does it deletes its contents.

```java
import java.io.PrintWriter;
import java.io.FileNotFoundException;

public class PrintTxt2File {
    void printDot(String fileContents, String fileName) {
        try(PrintWriter wr = new PrintWriter(new File(fileName))) {
            wr.print(fileContents);
        }
        catch(FileNotFoundException ex) {
            System.out.print("Unable to write file '"+fileName+"'");
        }
    }
}
```

Shipping Instructions

The work will be sent through the autolab platform. To submit, follow these steps:

- Zip the contents of the directory from your project**src/ce326/hw1/**, where are the three java files of this task. The resulting file must have a name **hw1.zip**.
- You are connecting toautolab (via VPN) and select the course**ECE326_2022 (S22)**and from that her task HW1.

Note: Similarity to this year's work or work from previous years results in automatic zeroing to the lesson of all involved.

- To submit your work, click on the option "I affirm that I have compiled with this course academic integrity policy..." and press submit. Then select the file**hw1.zip**where you created above.