TSU Desarrollo de Software Multiplataforma

Daniel Ivan Escobar Vasquez

Ruth Martinez Dominguez

Estructura de Datos - 4F

14 de Octubre del 2024

Contenido

Github

Practica 5

Listar palabras por letra Crea una aplicación para almacenar palabras (solicitadas al usuario), la aplicación debe de separar las palabras y dividirlas en listas clasificadas por la primera letra de la palabra es decir todas las palabras que coincidan en la primera letra.

Creamos una Clase que representa una palabra.

```
public class Letter {
    public LinkedList<String> wordList  ;
    public char character;
    public JLabel label;

    public Letter(char letter, JLabel label){
        this.wordList = new LinkedList<>();
        this.character = letter;
        this.label = label;
    }

    public void addWord(String newWorld){
        wordList.add(newWorld);
        String text = character + ": " + "";
        for(String word : wordList){
            text = text + ", " + word;
        }
        label.setText(text);
    }
}
```

El metodo `addWord()` muestra la letra ya la lista de palabras.

```
button.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent arg0) {

        String text = textField.getText(); // extract text from input field
        text = text.toLowerCase();
        String[] splitText = text.split("\\s+"); // split words from input field
```

```
        for (String word : splitText) {
            globalWordList.add(word);

            int letterIndex = letters.indexOf(word.charAt(0));

            if (letterIndex != -1) {
                Letter foundLetter = alphabet.get(letterIndex);
                foundLetter.addWord(word);

            } else {
                System.out.println("Character not found in the string.");
            }
        }
        textField.setText("");
    }
```

Se toma el texto, se dividen las palabras y cada una se añade a la respectiva letra.

Practica 6

Realiza una aplicación que lea una palabra e invierta el orden, por ejemplo, si recibes la palabra UTM el programa debe de invertirla y mostrar MTU.

Cramos una lista enlazada del tipo `Character`.

```
LinkedList<Character> letters = new LinkedList<>();
```

Aqui tomamos la palabra, creamos una lista enlazada y convierte cada letra en un nodo. Despues, recorremos la lista de forma invertida, retornando el ultimo elemento.

```
String text = word.getText(); // extract text from input field
text = text.toLowerCase();
for (char c : text.toCharArray()) {
    System.out.println(c);
    letters.add(c);
}

String reversedLetters = "";

for (int i = letters.size - 1; i >= 0; i--) {
    System.out.println(i);
    reversedLetters = reversedLetters + letters.get(i);
}
reversedText.setText(reversedLetters);
```

Practica 7

Un palíndromo es una palabra, número o frase que se lee igual hacia adelante que hacia atrás, algunos ejemplos de palíndromos son las palabras “ana”, arenera, arepera, anilina, Malayalam, Oruro, oso, radar, reconocer, rotor, salas, seres, somos, sometemos, entre otras. Realiza un programa que lea una palabra e indique si se trata de un palíndromo o no.

Creamos una lista enlazada de `Character` como en la practica anterior.

```
LinkedList<Character> letters = new LinkedList<>();
```

Realizamos los mismos pasos de la practica anterior: almacenar la entrada, cambiar el orden de las letras y remover los espacios. Finalmente, lo comparamos con la original. Si son iguales, significa que estamos ante un palindromo.

```
if(reversedLetters.equals(text)){
    System.out.println("ES PALINDROMO");
}
reversedText.setText(reversedLetters);
```