МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені Тараса Шевченка
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра програмних систем і технологій

Дисципліна
« Кросплатформне програмування,»

Лабораторна робота № 1
на тему:
"Статистичний аналіз тексту"

Виконав:	Безруков Андрій Миколайович	**Перевірив**:	Васильєв Олексій Миколайович
Група	ІПЗ-33	Дата перевірки	
Форма навчання	денна	Оцінка	
Спеціальність	121		
2024			

1. Постановка задачі

У рамках лабораторної роботи необхідно створити Java-застосунок, який виконує статистичний аналіз тексту з файлу. Зокрема, програма повинна:

1. Забезпечити вибір текстового файлу користувачем.
2. Зчитати вміст файлу та обчислити:
    - загальну кількість слів;
    - кількість оригінальних (унікальних) слів;
    - кількість речень;
    - кількість знаків пунктуації;
    - середню довжину слова;
    - середню довжину речення;
    - перші десять слів за частотою зустрічальності.
3. Продемонструвати роботу застосунку на операційних системах Windows та Linux.

---

2. Опис реалізації та програмний код

Нижче наведено вихідний код програми з поясненнями ключових фрагментів.

```
package lab1;
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Arrays;
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;

class lab1 {
  public static void main(String[] args) {
    try{
      //Open file
      Scanner scanner = new Scanner(System.in);
      System.out.println("Enter filepath");

      String filepath = scanner.nextLine();
      System.out.println("Filepath is: "+filepath);

      File file = new File(filepath);
```

```
        //read file
        Scanner myReader = new Scanner(file);
        while (myReader.hasNextLine()) {
            String data = myReader.nextLine();
            int countWords = data.split("\\s").length; //count words
            int countSentances = data.split("\\.").length; //count sentances
            int puntc = countPunctuation(data); //num of punctuation symbols

            int uniqueWords = countUniqueWords(data);//num of unique words

            double avgLen = countAvgWordLen(data, countWords);

            double avgLenS = countAvgSentenceLen(data, countSentances);


            //outputs
            System.out.println("words: "+countWords);
            System.out.println("Sentances: "+countSentances);
            System.out.println("Punctuation symbols: "+puntc);
            System.out.println("Number of unique words = " + uniqueWords);
            System.out.println("Average length of words = " + avgLen);
            System.out.println("Average length of sentences = " + avgLenS);
            countTopTen(data);
        }




        scanner.close();
        myReader.close();
    }catch(FileNotFoundException e) {
        System.out.println("An error occurred.");
        e.printStackTrace();
    }
}

public static int countPunctuation(String text){
    int number = 0;

    for (int i = 0; i < text.length(); i++)
    {
```

```
            //Checks whether given character is punctuation mark
            if(text.charAt(i) == '!' || text.charAt(i) == ',' || text.charAt(i) == ';' || text.charAt(i) == '.' ||
text.charAt(i) == '?' || text.charAt(i) == '-' ||  text.charAt(i) == '\'' || text.charAt(i) == '\"' || text.charAt(i) ==
':')
            {
                number++;
            }
        }
        return number;
    }

    public static int countUniqueWords(String text){
        //convert to lowercase to make counting case-insensitive
        text = text.toLowerCase();

        //remove punctuation and split by whitespace
        String[] words = text.replaceAll("[^a-zA-Z0-9\\s]", "").split("\\s+");

        //add all words to a HashSet (which only stores unique elements)
        Set<String> uniqueWords = new HashSet<>(Arrays.asList(words));

        //return the size of the set
        return uniqueWords.size();
    }

    public static double countAvgWordLen(String text, int count){
        int sum = 0;
        String[] words = text.split("\\s+");

        for (String word : words){
            double wordLen = word.length();
            sum += wordLen;
        }

        double avg = 0;
        if (count > 0){
            avg = sum/count;
        }

        return avg;
    }
```

```java
public static double countAvgSentenceLen(String text, int count){
    int sum = 0;
    String[] sent = text.split("\\.");

    for (String sentance : sent){
        double sentenceLen = sentance.length();
        sum += sentenceLen;
    }

    double avg = 0;
    if (count > 0){
        avg = sum/count;
    }

    return avg;
}

public static int countTopTen(String text){
    // Tokenize the input text into words and remove punctuation
    String[] words = text.replaceAll("[^a-zA-Z ]", "").toLowerCase().split("\\s+");

    // Count word frequencies
    String[] uniqueWords = new String[words.length];
    int[] wordCounts = new int[words.length];
    int uniqueWordCount = 0;

    for (String word : words) {
        boolean isUnique = true;

        for (int i = 0; i < uniqueWordCount; i++) {
            if (uniqueWords[i].equals(word)) {
                wordCounts[i]++;
                isUnique = false;
                break;
            }
        }

        if (isUnique) {
            uniqueWords[uniqueWordCount] = word;
            wordCounts[uniqueWordCount] = 1;
            uniqueWordCount++;
        }
```

```
    }

    // Sort word frequencies in descending order
    for (int i = 0; i < uniqueWordCount - 1; i++) {
        for (int j = i + 1; j < uniqueWordCount; j++) {
            if (wordCounts[i] < wordCounts[j]) {
                // Swap word frequencies
                int tempCount = wordCounts[i];
                wordCounts[i] = wordCounts[j];
                wordCounts[j] = tempCount;

                // Swap corresponding words
                String tempWord = uniqueWords[i];
                uniqueWords[i] = uniqueWords[j];
                uniqueWords[j] = tempWord;
            }
        }
    }

    // Display word frequencies
    System.out.println("\nFrequency of each word:");
    for (int i = 0; i < 10; i++) {
        System.out.println("- " + uniqueWords[i] + ": " + wordCounts[i]);
    }

    return 0;
  }
}
```

Пояснення коду:

- У методі main реалізовано вибір файлу, зчитування рядків та виклик функцій аналізу.
- countPunctuation рахує символи пунктуації з використанням набору знаків.
- countUniqueWords очищає рядок від небуквено-цифрових символів і підраховує унікальні слова через HashSet.
- countAvgWordLen і countAvgSentenceLen обчислюють середню довжину відповідно слів та речень.
- countTopTen визначає частоту використання кожного слова та виводить перші десять за спаданням.

3. Результати тестування

Програма успішно протестована в середовищах:

- **Windows 10**
- **Arch Linux**

Приклад запуску на Linux:

```
Enter filepath
/home/sad/labs/kyiv/kpp/lab1/lab1.txt
Filepath is: /home/sad/labs/kyiv/kpp/lab1/lab1.txt
words: 127
Sentances: 16
Punctuation symbols: 26
Number of unique words = 89
Average length of words = 5.0
Average length of sentences = 50.0

Frequency of each word:
- sit: 4
- amet: 4
- nulla: 4
- sed: 3
- auctor: 3
- at: 3
- elit: 3
- nec: 3
- tincidunt: 3
- odio: 3
```

4. Висновки

Виконана лабораторна робота з теми «Статистичний аналіз тексту» реалізована на Java. Створено консольний застосунок, який:

- читає вміст текстового файлу;
- обчислює базові статистичні показники (кількість слів, речень, пунктуації, унікальних слів);
- розраховує середню довжину слів та речень;
- виводить топ-10 найчастіших слів залежно від вмісту файлу.

Застосунок успішно функціонує в різних ОС (Windows, Linux), що підтверджує крос-платформенність рішення. Перспективи -подальшого розвитку: оптимізація обробки великих файлів, підтримка різних кодувань та мов аналізу.