МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

КАФЕДРА ИИТ

ОТЧЁТ

по лабораторной работе №2

Выполнил:

студент 3 курса

группы ПО-9

Тусюк Т.В.

Проверил:

Крощенко А.А.

**Цель работы:** приобрести базовые навыки работы с файловой системой в Java.

**Задание 1:**

9) Напишите программу, которая использует генерацию случайных чисел для создания предложений.

Программа должна использовать 4 массива строк, называемые noun (существительные),adjective (прилагательные), verb (глаголы) и preposition (предлоги).

Указанные массивы должны считываться из файла.

Программа должна создавать предложение, случайно выбирая слова из каждого массива в следующем порядке: noun, verb, preposition, adjective, noun.

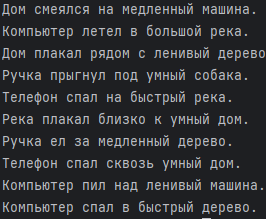
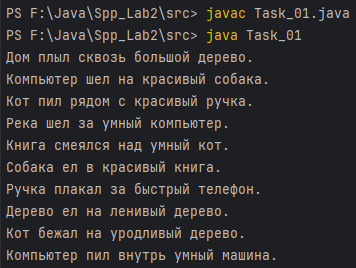
Слова должны быть разделены пробелами.

При выводе окончательного предложения, оно должно начинаться с заглавной буквы и заканчиваться точкой. Программа должна генерировать 20 таких предложений.

**Выполнение задания:**

import java.io.BufferedReader;  
import java.io.FileReader;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.Random;  
  
public class Task\_01 {  
  
 public static void main(String[] args) {  
 String nounsFile = "nouns.txt";  
 String adjectivesFile = "adjectives.txt";  
 String verbsFile = "verbs.txt";  
 String prepositionsFile = "prepositions.txt";  
   
 ArrayList<String> nouns = *readFile*(nounsFile);  
 ArrayList<String> adjectives = *readFile*(adjectivesFile);  
 ArrayList<String> verbs = *readFile*(verbsFile);  
 ArrayList<String> prepositions = *readFile*(prepositionsFile);  
   
 if (nouns.isEmpty() || adjectives.isEmpty() || verbs.isEmpty() || prepositions.isEmpty()) {  
 System.*err*.println("Ошибка: Не удалось найти необходимые данные в файлах.");  
 return;  
 }  
   
 Random random = new Random();  
 for (int i = 0; i < 20; i++) {  
 String sentence = *generateSentence*(nouns, adjectives, verbs, prepositions, random);  
 System.*out*.println(sentence);  
 }  
 }  
   
 private static ArrayList<String> readFile(String fileName) {  
 ArrayList<String> words = new ArrayList<>();  
 try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 words.add(line.trim());  
 }  
 } catch (IOException e) {  
 System.*err*.println("Ошибка при чтении файла " + fileName + ": " + e.getMessage());  
 }  
 return words;  
 }  
   
 private static String generateSentence(ArrayList<String> nouns, ArrayList<String> adjectives,  
 ArrayList<String> verbs, ArrayList<String> prepositions, Random random) {  
 String noun = *getRandomElement*(nouns, random);  
 String adjective = *getRandomElement*(adjectives, random);  
 String verb = *getRandomElement*(verbs, random);  
 String preposition = *getRandomElement*(prepositions, random);  
 String secondNoun = *getRandomElement*(nouns, random);  
  
 // Формирование предложения  
 String sentence = noun + " " + verb + " " + preposition + " " + adjective + " " + secondNoun + ".";  
 // Преобразование первой буквы в заглавную  
 sentence = sentence.substring(0, 1).toUpperCase() + sentence.substring(1);  
 return sentence;  
 }  
   
 private static String getRandomElement(ArrayList<String> list, Random random) {  
 int index = random.nextInt(list.size());  
 return list.get(index);  
 }  
}

**Результат:**

****

**Задание 2:**

9) Утилита join объединяет строки двух упорядоченных текстовых файлов на основе наличия

общего поля. По своему функционалу схоже с оператором JOIN , используемого в языке SQL

для реляционных баз данных, но оперирует с текстовыми файлами.

Команда join принимает на входе два текстовых файла и некоторое число аргументов. Если не

передаются никакие аргументы командной строки, то данная команда ищет пары строк в двух

файлах, обладающие совпадающим первым полем (последовательностью символов, отличных

от пробела), и выводит строку, состоящую из первого поля и содержимого обоих строк.

Ключами -1 или -2 задаются номера сравниваемых полей для первого и второго файла, соот-

ветственно. Если в качестве одного из файлов указано – (но не обоих сразу!), то в этом случае

вместо файла считывается стандартный ввод.

Формат использования:

join [-1 номер\_поля] [-2 номер\_поля] файл1 файл2 [файл3]

Параметры:

• - 1 fiedl\_num Задает номер поля в строке для первого файла, по которому будет выпол-

няться соединение.

• - 2 field\_num Задает номер поля в строке для второго файла, по которому будет выпол-

няться соединение.

Аргументы:

• файл1, файл2 – входные файлы

• файл3 – выходной файл, куда записывается результат работы программы.

Примеры использования:

Пусть задан файл 1.txt со следующим содержимым:

1 abc

2 lmn

3 pqr

и файл 2.txt со следующим содержимым:

1 abc

3 lmn

9 opq

Тогда, выполнение команды join 1.txt 2.txt даст следующий результат:

1 abc abc

3 pqr lmn

Поскольку в обоих файлах есть строки, чьё первое поле совпадает (1, 3), выполнение команды

join -1 2 -2 2 1.txt 2.txt даст результат

abc 1 1

lmn 2 3

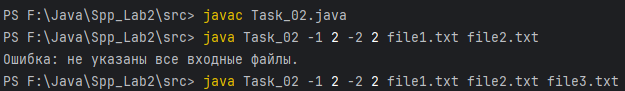
поскольку теперь сравнение выполняется по 2-му полю для первого и второго файла соответ-

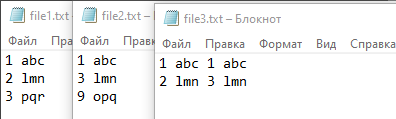
ственно.

**Выполнение задания:**

import java.io.BufferedReader;  
import java.io.BufferedWriter;  
import java.io.FileReader;  
import java.io.FileWriter;  
import java.io.IOException;  
import java.util.HashMap;  
import java.util.Map;  
  
public class Task\_02 {  
  
 public static void main(String[] args) {  
 if (args.length < 3) {  
 System.*err*.println("Использование: join [-1 номер\_поля] [-2 номер\_поля] файл1 файл2 файл3");  
 return;  
 }  
  
 int field1 = 1;  
 int field2 = 1;  
 String file1 = null;  
 String file2 = null;  
 String file3 = null;  
  
 int i = 0;  
 try {  
 while (i < args.length) {  
 if (args[i].equals("-1")) {  
 field1 = Integer.*parseInt*(args[i + 1]);  
 i += 2;  
 } else if (args[i].equals("-2")) {  
 field2 = Integer.*parseInt*(args[i + 1]);  
 i += 2;  
 } else if (file1 == null) {  
 file1 = args[i];  
 i++;  
 } else if (file2 == null) {  
 file2 = args[i];  
 i++;  
 } else {  
 file3 = args[i];  
 i++;  
 }  
 }  
 } catch (NumberFormatException e) {  
 System.*err*.println("Ошибка: номер поля должен быть целым числом.");  
 return;  
 }  
  
 if (file1 == null || file2 == null || file3 == null) {  
 System.*err*.println("Ошибка: не указаны все входные файлы.");  
 return;  
 }  
  
 try (BufferedReader reader1 = new BufferedReader(new FileReader(file1));  
 BufferedReader reader2 = new BufferedReader(new FileReader(file2));  
 BufferedWriter writer = new BufferedWriter(new FileWriter(file3))) {  
  
 Map<String, String> map = new HashMap<>();  
  
 String line;  
 while ((line = reader1.readLine()) != null) {  
 String[] parts = line.split("\\s+", Math.*max*(field1, 2));  
 if (parts.length >= field1) {  
 map.put(parts[field1 - 1], line);  
 }  
 }  
  
 while ((line = reader2.readLine()) != null) {  
 String[] parts = line.split("\\s+", Math.*max*(field2, 2));  
 if (parts.length >= field2 && map.containsKey(parts[field2 - 1])) {  
 String resultLine = map.get(parts[field2 - 1]) + " " + line;  
 writer.write(resultLine);  
 writer.newLine();  
 }  
 }  
 } catch (IOException e) {  
 System.*err*.println("Ошибка при чтении/записи файла: " + e.getMessage());  
 }  
 }  
}

**Результат:**

****

****

**Вывод:** приобрел практические навыки обработки параметров командной строки, закрепил базовые знания языка программирования Java при решении практических задач.