Programowanie w języku JAVA

Laboratorium 4

Marcin Godfryd Grupa 31

1. MyLinkedList.java

import java.util.Iterator;  
import java.util.NoSuchElementException;  
  
public class MyLinkedList<T> implements Iterable<T> {  
 private Node<T> head;  
  
 private static class Node<T> {  
 T value;  
 Node<T> next;  
  
 Node(T value) {  
 this.value = value;  
 this.next = null;  
 }  
 }  
  
 public void add(T value) {  
 Node<T> newNode = new Node<>(value);  
 if (head == null) {  
 head = newNode;  
 } else {  
 Node<T> current = head;  
 while (current.next != null) {  
 current = current.next;  
 }  
 current.next = newNode;  
 }  
 }  
  
 @Override  
 public Iterator<T> iterator() {  
 return new Iterator<>() {  
 Node<T> current = head;  
  
 @Override  
 public boolean hasNext() {  
 return current != null;  
 }  
  
 @Override  
 public T next() {  
 if (!hasNext()) {  
 throw new NoSuchElementException();  
 }  
 T value = current.value;  
 current = current.next;  
 return value;  
 }  
 };  
 }  
  
 public boolean contains(T value) {  
 Node<T> current = head;  
 while (current != null) {  
 if (current.value.equals(value)) {  
 return true;  
 }  
 current = current.next;  
 }  
 return false;  
 }  
  
 public static void main(String[] args) {  
 MyLinkedList<String> list = new MyLinkedList<>();  
 list.add("Jeden");  
 list.add("Dwa");  
 list.add("Trzy");  
  
 System.*out*.println("Elementy listy:");  
 for (String s : list) {  
 System.*out*.println(s);  
 }  
  
 System.*out*.println("Czy lista zawiera 'Jeden'? " + list.contains("Jeden"));  
 System.*out*.println("Czy lista zawiera 'Cztery'? " + list.contains("Cztery"));  
 }  
}

1. Job.java

import java.util.PriorityQueue;  
  
public class Job implements Comparable<Job> {  
 private final String description;  
 private final int priority;  
  
 public Job(String description, int priority) {  
 this.description = description;  
 this.priority = priority;  
 }  
  
 @Override  
 public int compareTo(Job other) {  
 return Integer.*compare*(other.priority, this.priority);  
 }  
  
 @Override  
 public String toString() {  
 return "Job{" +  
 "description='" + description + '\'' +  
 ", priority=" + priority +  
 '}';  
 }  
  
 public static void main(String[] args) {  
 PriorityQueue<Job> jobQueue = new PriorityQueue<>();  
  
 jobQueue.add(new Job("Job 1", 5));  
 jobQueue.add(new Job("Job 2", 3));  
 jobQueue.add(new Job("Job 3", 6));  
 jobQueue.add(new Job("Job 4", 1));  
 jobQueue.add(new Job("Job 5", 2));  
 jobQueue.add(new Job("Job 6", 77));  
 jobQueue.add(new Job("Job 7", 7));  
 jobQueue.add(new Job("Job 8", 10));  
 jobQueue.add(new Job("Job 9", 4));  
 jobQueue.add(new Job("Job 10", 9));  
  
 while (!jobQueue.isEmpty()) {  
 System.*out*.println(jobQueue.poll());  
 }  
 }  
}

1. NumberGenerator.java

import java.util.Random;  
import java.util.stream.IntStream;  
  
public class NumberGenerator {  
 public static void main(String[] args) {  
 Random random = new Random();  
 int[] numbers = new int[10];  
  
 for (int i = 0; i < 10000; i++) {  
 int number = random.nextInt(10);  
 numbers[number]++;  
 }  
  
 System.*out*.println("Wystąpienia w kolejności naturalnej:");  
 for (int i = 0; i < numbers.length; i++) {  
 System.*out*.println(i + ": " + numbers[i]);  
 }  
  
 System.*out*.println("\nWystąpienia w kolejności malejącej:");  
 IntStream.*range*(0, numbers.length)  
 .boxed()  
 .sorted((i, j) -> numbers[j] - numbers[i])  
 .forEach(i -> System.*out*.println(i + ": " + numbers[i]));  
 }  
}

1. NumberGeneratorStreamsAPI.java

import java.util.Map;  
import java.util.Random;  
import java.util.function.Function;  
import java.util.stream.Collectors;  
  
public class NumberGeneratorStreamsAPI {  
 public static void main(String[] args) {  
 Map<Integer, Long> numbers = new Random().ints(10000, 0, 10)  
 .boxed()  
 .collect(Collectors.*groupingBy*(Function.*identity*(), Collectors.*counting*()));  
  
 System.*out*.println("Wystąpienia w kolejności naturalnej:");  
 numbers.entrySet().stream()  
 .sorted(Map.Entry.*comparingByKey*())  
 .forEach(entry -> System.*out*.println(entry.getKey() + ": " + entry.getValue()));  
  
 System.*out*.println("\nWystąpienia w kolejności malejącej:");  
 numbers.entrySet().stream()  
 .sorted(Map.Entry.<Integer, Long>*comparingByValue*().reversed()  
 .thenComparing(Map.Entry.*comparingByKey*()))  
 .forEach(entry -> System.*out*.println(entry.getKey() + ": " + entry.getValue()));  
 }  
}

1. FIleAnalyzer.java
2. import java.nio.charset.StandardCharsets;  
   import java.nio.file.Files;  
   import java.nio.file.Paths;  
   import java.util.Comparator;  
   import java.util.List;  
     
   public class FileAnalyzer {  
    public static void main(String[] args) {  
    String filePath = "./sales\_data\_sample.csv";  
     
    try {  
    List<String> topTransactions = Files.*lines*(Paths.*get*(filePath), StandardCharsets.*ISO\_8859\_1*)   
    .skip(1)  
    .map(line -> line.split(","))  
    .filter(tokens -> tokens.length > 4 && *isNumeric*(tokens[4]))  
    .sorted(Comparator.*comparingDouble*(tokens -> -Double.*parseDouble*(tokens[4])))  
    .limit(5)  
    .map(tokens -> String.*join*(", ", tokens))  
    .toList();  
     
    topTransactions.forEach(System.*out*::println);  
    } catch (Exception e) {  
    e.printStackTrace();  
    }  
    }  
     
    private static boolean isNumeric(String str) {  
    try {  
    Double.*parseDouble*(str);  
    return true;  
    } catch(NumberFormatException e) {  
    return false;  
    }  
    }  
   }