Practice6

На странице представлена информация по практическому заданию №6.

Заглушка проекта

Stub проекта находится ~~в репозитории: /examples/Practice6Stub~~ здесь: тынц.

Задание 1

Разработать приложение, которое считывает текст из консольного ввода и выводит слова в порядке убывания их частоты появления в тексте (при совпадении частот порядок - лексикографичекий). Решить задачу с применением ООП подхода. Класс Word, содержит строковое поле word и целое поле frequency, контейнер WordContainer агрегирует объекты Word. При использовании контейрнерных классов из ядра грамотно реализовать методы Word#equals / Word#hashCode / Word#compareTo (если они будут нужны).

Если вход такой: asd asdf asd asdf asdf 43 asdsf 43 43 434

То выход должен быть таким:

43 : 3
asdf : 3
asd : 2
434 : 1
asdsf : 1

Демонстрационный метод Par1.main должен содержать моделирование консольного ввода.

Задание 2

Написать программу, которая моделирует следующий процесс. В кругу стоят n человек, пронумерованных от 0 до n-1. При ведении счета по кругу вычеркивается каждый k-й человек (0 < k < n), пока не останется один. Решить задачу двумя способами - с использованием класса ArrayList и с использованием класса Linked List. Определить и сравнить время выполнения каждой из двух программ на одинаковых входных данных (задавать в Part2.main).

Задание 3

Реализовать класс, моделирующий работу n-местной автостоянки. Машина подъезжает к определенному месту и едет вправо, пока не встретится свободное место. Класс должен поддерживать методы, обслуживающие приезд и отъезд машины. Определить метод, который выводит в консоль текущее состояние стоянки. Продемонстрировать работу приложения (Part3.main).

Задание 4

Реализовать класс Graph, представляющий собой неориентированный граф. В конструкторе класса передается количество вершин в графе. Методы должны поддерживать быстрое добавление и удаление ребер. Продемонстрировать работу приложения (Part4.main)

.

Задание 5

Создать generic класс Tree, который реализует структуру данных "двоичное дерево поиска". Контейнерные классы не использовать.

Tree.java

```
public class Tree<E extends Comparable<E>> {
    // добавляет элемент в контейнер
    // если в контейнере есть элемент равный по compareTo добавляемому,
    // то добавления не происходит и метод возвращает false
    // в противном случае элемент попадает в контейнер и метод возвращает true
    // первый добавляемый элемент становится корнем дерева
    public boolean add(E element) {...}

    // добавляет все элементы из массива в контейнер (вызов в цикле метода add, см. выше)
    public void add(E[] elements) {...}

    // удаляет элемент из контейнера
    // если удаляемого элемента в контейнере нет, то возвращает false
    // в противном случае удаляет элемент и возвращает true
    // ВАЖНО! при удалении элемента дерево не должно потерять свойства бинарного дерева поиска
    public boolean remove(E element) {...}

    // распечатывает дерево, так чтобы было видно его древовидную структуру, см. ниже пример
    public void print() {...}

    // вложенный класс, объекты этого класса составляют дерево
    private static class Node<E> {...}
}
```

Код

```
Tree<Integer> tree = new Tree<>();

System.out.println(tree.add(3));
System.out.println(tree.add(3));

System.out.println("~~~~~~~");
tree.add(new Integer[] {1, 2, 5, 4, 6, 0});
tree.print();

System.out.println("~~~~~~~");
System.out.println(tree.remove(5));
System.out.println(tree.remove(5));

System.out.println("~~~~~~~");
tree.print();
```

Вывод

```
true
false
~~~~~~~
      0
   1
      2
3
      4
   5
      6
~~~~~~~
true
false
~~~~~~~
      0
   1
      2
6
      4
   5
```

Задание 6

Реализовать консольное приложение, которое анализирует текст. Формат входных параметров командной строки для приложения (программа должна понимать как короткие так и длинные опции):

- -i (или --input) путь к входному файлу;
- -t (или --task) наименование подзадачи.

Примеры параметров командной строки для запуска приложения

-i input.txt -t frequency
--input input.txt --task length

Подзадач всего три (для их представления можно использовать например enum): frequency, length, duplicates.

Part 1 (frequency)

Во входном файле найти три слова, которые встречаются наиболее часто, и распечатать их отсортированными по алфавиту в обратном порядке в формате: слово ==> частота

Пример вывода

```
panda ==> 15
ezhik ==> 20
apple ==> 19
```

Part 2 (length)

Во входном файле найти три самых длинных слова и распечатать их в формате: слово ==> количество букв в слове. Список должен быть отсортирован по убыванию количества букв в слове.

Пример вывода

```
anesthetist ==> 11
kitchen ==> 7
bird ==> 4
```

Part 3 (duplicates)

Во входном файле найти первые три слова, которые имеют дубликаты, и напечатать их инверсию в верхнем регистре.

Пример вывода

```
ADNAP
TAC
ENIGREBUA
```

Задание 7

Написать класс Range, который бы представлял собой промежуток чисел [n, m], где n < m. Класс должен реализовывать интерфейс Iterable. Итератор реализовать таким образом, чтобы он проходил он начала до конца промежутка. В конструктор передавать дополнительный параметр reverse.

Пример.

Промежуток - [3, 10]

reverse = true - итератор проходит от начала до конца промежутка 3, 4, 5, 6, 7, 8, 9, 10

reverse = false - итератор проходит от конца до начала промежутка (т.е. в обратном порядке) 10, 9, 8, 7, 6, 5, 4, 3