Sofia University
Department of Mathematics and Informatics

Course : Applied OO Programming part 1
Date: May 19, 2020
Student Name:

Lab No. 13a

Инструкции- задачите да се решават, чрез използване на Collections библиотеката на Java

Задача 1.

Напишете class **ArrayListTest** за тестване на следните методи.

a) метод, който връща най- големия елемент в двумерен масив.

```
public static <E extends Comparable<E>> E max(E[][] list)
```

b) метод, който разбърква по произволен начин (**shuffles)** елементите .на **ArrayList**:

```
public static <E> void shuffle(ArrayList<E> list)
```

c) метод, който връща най- големия елемент на **ArrayList**:

```
public static <E extends Comparable<E>> E
max(ArrayList<E> list)
```

d) метод, който връща нов **ArrayList**. Новият списък на съдържа само различните елементи от подадения като параметър списък .

```
public static <E> ArrayList<E>
removeDuplicates(ArrayList<E> list)
```

Задача 2.1

Напишете class **ArrayListTest** който има **ArrayList<String> list**.

a) Напишете следния метод

```
public void sort()
```

за сортиране на елементите на данната **list** в намаляващ ред като използвате анонимен клас, производен на **Comparator**.

b) Напишете следния метод

```
public void sortByFrequency()
```

за сортиране на елементите на данната **list** в намаляващ ред на честотата, с която се срещат в списъка ред като използвате анонимен клас, производен на **Comparator**.

c) Напишете следния метод

```
public static <T extends Comparable<? super T>>
                                                    T
removeMax(List<T> someList)
{ ... }
```

за изтриване на най- големия елемент, където методът връща референция към изтрития елемент. Когато **someList** е празен **removeMax**() връща **null**

d) Напишете следния метод

```
public void  getNames()
```

за прочитане на имена на студенти от стандартен вход и записване в списък само различните данната **list**.

e) Напишете следния метод

```
public void copyTo(ArrayList<String> str)
```

който копира данната **list** в **str.** Приложете **Принципа Get- Put** за параметъра за тип на **ArrayList<String> str**

f) Напишете следния метод

```
public String toStringDescending()
```

който връща String с елементите на данната **list в** намаляващ ред. Елементите да са разделени с двуеточие.

g) Напишете следния метод

```
public static List<?>
        append(List<? extends Number> numberSrc,
                                      Integer     n)
```

който добавя **n** към **numberSrc**. Правилно ли е приложен Принципът GET- PUT в този случай? Има ли нужда от промяна в декларацията на метода? Тествайте изпълнението на този метод като

```
append(new ArrayList<Integer> src, 10)
```

Тествайте изпълнението на методите (a- f) и (g) в **main**() метода на `class` **ArrayListTest.**

Задача 2.2

Да предположим, че данната **list** в class **ArrayListTest** от Задача 2.1 е списък съдържащ низовете `Лили, Мария, Георги, Илия, Цвета, Георги.`

a) .Какъв ще бъде резултата от изпълнението на следните команди?

```
Iterator<String> nameIterator = list.iterator();
System.out.println(nameIterator.next());
nameIterator.next();
System.out.println(nameIterator.next());
nameIterator.remove();
System.out.println(nameIterator.next());
System.out.println(nameList);
```

b) Нека **`list`** и `nameIterator` са тези дефинирани в условие (a) Напишете команди за извеждане на всички низове в **`list`** от **последния** до първия
c) Нека **`list`** и `nameIterator` са тези дефинирани в условие(a). Напишете команди използващи `nameIterator` за изтриване на всички елементи от списъка
d) Нека `aList` и `bList` са `ArrayList` от `String` елементи. Използвайте два итератора за намиране и извеждане на общите елементи в двата списъка без да променяте съдържанието на изходните списъци `aList` и `bList`
e) Нека **`list`** е данната на class **`ArrayListTest`** инициализиран, както в условие(a). Използвайте `ListIterator<String> nameIterator` , за да напишете командите за добавяне на низа Симеон след първото появяване на Георги в **`list`**

Задача 3.

Променете програмата на **Fig. 19.20** да преброява броя на срещане на всяка буква, вместо на всяка дума. Например, низът `"HELLO THERE"` се състои от 2 букви `H`, три `E`, две `L`, една `O`, една `T` и едно `R`. Изведете крайните резултати

Упътване: Използвайте приложения код на `Fig. 19.20` (`fig19_20.rar`)

Задача 4.

При извеждане на крайните резултати в **Fig. 19.17** (`PriorityQueueTest`) се вижда, че `PriorityQueue` сортира `Double` елементите във възходящ ред. Пренапишете `Fig. 19.17` така че да сортира `Double` елементите в **низходящ** ред.

Упътване: Използвайте приложения код на `Fig. 19.17` (`fig19_17.rar`)

Задача 5.

Напишете програма, която създава `Set s` от `Integer` обекти със стойности `{7, 12, 15, 19, 53, 68, 3, 33, 57, 45, 25}` и един `ArrayList t` от `Integer` обекти `{22, 9, 15, 42, 53, 79, 3, 33, 97, 45, 25}`.
Напишете методи, които връщат множеството `Set` от

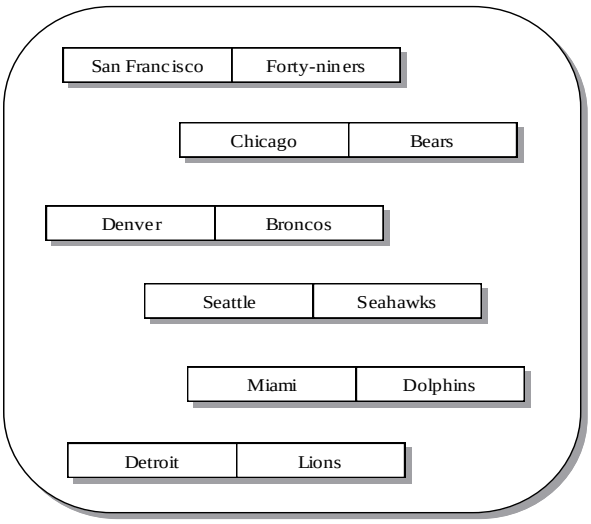
- Обединението на елементите от `s` и `t` (използвайте метод `addAll`()).
- Сечението от елементите от `s` и `t` (използвайте метод `retainAll`()).
- Разликата от на елементите от `s` и `t` (използвайте метод `removeAll`()).
- Разликата от на елементите от `s` и `t` (използвайте метод `removeAll`()).

Изведете списъка с получените множества от елементи, както и списъците с дадените стойности

Напишете метод
`public Set<Integer> sortDescending( Set<Integer> set)`
който връща т множеството `set` сортирано в низходящ ред. Тествайте изпълнението на този метод с множеството `s` като изведете на стандартен изход резултата от изпълнението на метода.

Задача 6

Напишете приложение което създава `TreeMap<String, String> teams`, която съдържа следните двойки (ключ- стойност) от град и клуб на отбор от този град.



Изпълнете следните операции.

a) Изведете броят на елементите в изображението `TreeMap<String, String> teams` и името на отбора от `Chicago`.
b) Променете името на отбора от `San Francisco` да бъде `"Niners"`.
c) Отговорете н авъпроса, дали `San Diego` има отбор въведен в съответствието `TreeMap<String, String> teams` .
d) Изтрийте `Denver` от съответствието `TreeMap<String, String> teams`.
e) Въведете отбора `Cowboys` на `Dallas` в съответствието `TreeMap<String, String> teams` .
f) Изведете на стандартен изход полученото съответствие `TreeMap<String, String> teams`.

g) Изведете на стандартен изход ключовете и стойностите на полученото съответствие `TreeMap<String, String> teams`, **сортирани в низходящ ред на стойностите** т.е. имената на клубовете на отборите, заедно с градовете на тези клубове.

Задача 7

Използвайки итератор, имплементирайте **`static`** версия на метода **`addAll()`** на интерфейс **`Collection`** като обединение на **`Set`** и елементите на произволна колекция **.** Методът връща **`Set`**.

```
public static <T> Set<T> addAll(Set<T> s,
                                Collection<? extends T> c)
{ …. }
```

Упътване: Декларирайте итератоа като

```
Iterator<? extends T> iter = c.iterator();
```

Тествайте метода като създадете **`TreeSet<String> set c`** елементи на зададен масив **`String[] strArrA`** и допълнително **`ArrayList<String> list`**, чиито елементи са друг зададен масив **`strArrB`**.

Изпълнете **`addAll`**() и създайте **`TreeSet`** който да е обединение на създадения **`set`** и **`list`**. Изведете резултата от изпълнението на метода.

Примерни данни:

```
String strArrA[] = {"dog", "cat", "tiger", "pig"},
strArrB[] = {"frog", "dog", "monkey", "pig", "snake"};
```

Задача 8

Напишете програма, която връща елемента(елементите), които се срещат най-често в масив от цели числа. Използвайте `ArrayList` за преброяване на броя пъти, който се среща всеки елемент от масива. Накрая изведете най- често срещания елементи(елементи) и броя пъти, които се среща(срещат) в масива.

Напишете версия на програмата, която да работи с масив от String, вместо с цели числа.

Използвайте следния сорс код.

```
import java.util.ArrayList;

public class MostFrequentElement {

/**
Given an array of ints, the program finds and prints the most
frequently occurring element and its number of occurrences.
If there is more than one such element, any one of them may be
printed.
Assume that the given array contains at least one element.
**/
public static void main(String [] args) {
     // in this example 1 is the most frequent element,
```

```
      // it appears 7 times:
      int [] elements =
      {1, 3, 4, 1, 5, 2, 3, 6, 6, 6, 4, 1, 2, 6,
       2, 3, 1, 2, 1, 5, 5, 1, 1, 5, 4};

        // Your code goes


      // fill in the appropriate results:
      System.out.println("The most frequent element " +
      "occurs " +  " times");
     }
}
```

Задача 9.

`class Stack` е реализиран в `Collections` библиотеката на Java като производен на class `Vector`. Това води до определени неудобства. Напишете class `Stack` , който има class `Vector (от Collections` библиотеката на Java) и реализирайте интерфейс `List<T>` в този клас като не забравите да използвате подходящи параметри за тип. `class Stack` трябва да има и конструктор с аргумент реферирящ `Collection` обект. Методите от интерфейс `List<T>` , които не съответстват на дефиницията на `class Stack` да се реализират в съответствие с тази дефиниция. Например, може да изтрием `Object o` от `Stack` с метода
`public boolean remove(Object o)` на интерфейс `List<T>`
само, ако първият елемент на `Stack` съвпада с `o`. В противен случай, не изтриваме елемента `o` и връщаме `false`.
Напишете програма за тестване на методите на class `Stack`:

- `push()`
- `pop()`
- `peek()`
- `toString()`
- Методите на interface `List<T>`