**Автоматизация отчётности о продажах**

Интернет-магазин "Платья и Сумки" быстро расширяется, и его создатели заинтересованы в подробной аналитической отчётности о продажах товара. К сожалению, первая версия магазина была сделана очень криво, поэтому данные хранятся в системе в виде, плохо подходящем для обработки. Так, каждая запись о продаже представляет собой строку формата

название-товара количество-проданных-штук

например:

платье1 5

сумка32 2

платье1 1

сумка23 2

сумка128 4

Названия товаров могут повторяться.

Ваша задача: сгруппировать продажи по названиям товаров, расположив в результирующем списке товары, отсортированные по количеству продаж. Если эти количества для каких-то товаров совпадут, названия товаров должны следовать в порядке лексикографического возрастания.

Например, вышеприведённый пример преобразуется в такой результат:

платье1 6

сумка128 4

сумка23 2

сумка32 2

Функция

string [] ShopOLAP(int N, string [] items)

получает на вход N >= 1 строк о товарах в вышеприведённом формате, и выдаёт массив длиной M <= N, содержащий сводку по продажам в сгруппированном виде.

[как постить решение](https://skillsmart.ru/algo/lvl1/how.html)

предварительное решение:

отделяем цифры после пробела, сортируем по кол-ву продаж,

потом займемся названиями

отделение цифр после пробела и вывод как INT

//method to extract numbers after whitespace as an int  
static int amount (String inputSubstr){  
 int finalAmount = 0;  
 int spaceIndex = 0;  
  
 spaceIndex = inputSubstr.indexOf(' ');  
 String subs = inputSubstr.substring(spaceIndex + 1);  
 finalAmount = Integer.*parseInt*(subs);  
  
 return finalAmount;  
}

Далее сортируем по количеству продаж

// method to sort string array descending by numbers after whitespace  
static String[] bubbleSort(String[] inputStr3) {  
 List<String> list1 = new ArrayList<>();  
  
 for (int i = 0; i < inputStr3.length; i++){  
 list1.add(inputStr3[i]);  
 }  
 list1.add("!!!temp!!!");  
 int n = list1.size();  
  
 for (int i = 0; i < (n - 1); i++) {  
  
 for (int j = 1; j < (n - i - 1); j++) {  
  
 if (*amount*(list1.get(j - 1)) < (*amount*(list1.get(j)))) {  
  
 list1.set(n - 1, list1.get(j - 1));  
 list1.set(j - 1, list1.get(j));  
 list1.set(j, list1.get(n - 1));  
 }  
 }  
 }  
 list1.remove(n - 1);  
  
 String[] sortedStringArr = list1.toArray(new String[0]);  
 return sortedStringArr;  
}

Например строка

String[] test = {"bag31 22", "bag3 22", "bag132 444", "bag23 22", "bag33 444", "bags24 22", "bags33 444", "bag24 22"};

Отсортируется следующим образом:

bag132 444, bag33 444, bags33 444, bag31 22, bag3 22, bag23 22, bags24 22, bag24 22

далее группируем все одинаковые кол-ва продаж (444, 22, …)

по отдельным столбцам массива:

[bag132 444, bag33 444, bags33 444]

[bag31 22, bag3 22, bag23 22, bags24 22, bag24 22]

И так далее

А внутри этих массивов уже сортируем их по лексикографическому возрастанию

ТАДА !!!!!!!!!!!!!! япростогений!!!

Следующий метод не нужен, сохранил на всякий случай

//method to find index of occurrence of first digit in the string  
static int firstDigit (String inputStr2){  
 int index = -1;  
  
 for (int d = 0; d < inputStr2.length(); d++){  
 if (Character.*isDigit*(inputStr2.charAt(d))) {  
 index = d;  
 break;  
 }  
 }  
 return index;  
}

|  |
| --- |
| / A sample Java program to sort a subarray  // using Arrays.sort().  import java.util.Arrays;    public class SortExample  {      public static void main(String[] args)      {          // Our arr contains 8 elements          int[] arr = {13, 7, 6, 45, 21, 9, 2, 100};            // Sort subarray from index 1 to 4, i.e.,          // only sort subarray {7, 6, 45, 21} and          // keep other elements as it is.          Arrays.sort(arr, 1, 5);            System.out.printf("Modified arr[] : %s",                            Arrays.toString(arr));      }  } |

Output:

Modified arr[] : [13, 6, 7, 21, 45, 9, 2, 100]

Рабочее решение:

import java.util.\*;  
  
public class Sales {  
  
 static String [] ShopOLAP(int N, String [] items){  
 String [] finalResult = *bubbleSort*(items);  
 //sorting same sales groups (same sales numbers after whitespace)  
 int offset = 0;  
 for (int i = 1; i < finalResult.length; i++){  
 if (*amount*(finalResult[i]) != *amount*(finalResult[i - 1])){  
 Arrays.*sort*(finalResult, offset, i);  
 offset = i;  
 }  
 }  
 return finalResult;  
 }  
  
 //method to extract numbers after whitespace as an int  
 static int amount (String inputSubstr){  
 int finalAmount = 0;  
 int spaceIndex = 0;  
  
 spaceIndex = inputSubstr.indexOf(' ');  
 String subs = inputSubstr.substring(spaceIndex + 1);  
 finalAmount = Integer.*parseInt*(subs);  
  
 return finalAmount;  
 }  
  
 // method to sort string array descending by numbers after whitespace  
 static String[] bubbleSort(String[] inputStr3) {  
 List<String> list1 = new ArrayList<>();  
  
 for (int i = 0; i < inputStr3.length; i++){  
 list1.add(inputStr3[i]);  
 }  
 list1.add("!!!temp!!!");  
 int n = list1.size();  
  
 for (int i = 0; i < (n - 1); i++) {  
  
 for (int j = 1; j < (n - i - 1); j++) {  
  
 if (*amount*(list1.get(j - 1)) < (*amount*(list1.get(j)))) {  
  
 list1.set(n - 1, list1.get(j - 1));  
 list1.set(j - 1, list1.get(j));  
 list1.set(j, list1.get(n - 1));  
 }  
 }  
 }  
 list1.remove(n - 1);  
  
 String[] sortedStringArr = list1.toArray(new String[0]);  
 return sortedStringArr;  
 }  
  
  
 public static void main(String[] args) {  
 String[] test = {"bag31 22", "bag3 22", "bag132 444", "bag23 22", "bag33 444", "bags24 11", "bags33 444", "bag24 22"};  
 int n = 8;  
  
  
  
System.*out*.println(Arrays.*toString*(*ShopOLAP*(n, test)));  
  
  
  
 }  
}

решение выше неполное, т.к. мы забыли обьединить одинаковые наименования товаров, т.е. bag23 22 и bag23 32 должно быть -> bag23 55

для этого нам надо выделить все названия через след метод:

//method to extract item name (substring before whitespace)  
static String itemName (String inputStr1){  
 int spaceIndex1 = 0;  
 spaceIndex1 = inputStr1.indexOf(' ');  
 String subs = inputStr1.substring(0, spaceIndex1);  
 return subs;  
}

Дальше пройтись по массиву и суммировать кол-во продаж одноименных товаров

Финальное решение поэтапно:

Данный метод находит одинаковые елементы по именнам и суммирует кол-во их продаж

static String[] removeDuplicates(String[] inputString) {  
 int sum = 0;  
  
 List<String> list2 = new ArrayList<>();  
 for (int i = 0; i < inputString.length; i++){  
 list2.add(inputString[i]);  
 }  
  
 //creating array for duplicate item names  
 ArrayList<Integer> duplicates = new ArrayList<>();  
 int length = list2.size();  
  
 //outer loop for name check  
 for (int j = 0; j < length; j++) {  
 sum = *amount*(list2.get(j));  
  
  
 for (int i = j + 1; i < length; i++) {  
 //calculating sum of sales of duplicate names  
 if (*itemName*(list2.get(j)).equals(*itemName*(list2.get(i)))){  
 duplicates.add(i);  
 sum = sum + *amount*(list2.get(i));  
 }  
 }  
  
 if (duplicates.size() == 1) {  
 int a = duplicates.get(0);  
 list2.remove(a);  
 length--;  
 //replacing current item with same name + total sales  
 list2.set(j, *itemName*(list2.get(j)) + " " + sum);  
 }  
  
 if (duplicates.size() > 1) {  
 int dupSize = duplicates.size();  
 for (int r1 = dupSize - 1; r1 >= 0; r1--){  
 int a = duplicates.get(r1);  
 list2.remove(a);  
 }  
 length = length - dupSize;  
 //replacing current item with same name + total sales  
 list2.set(j, *itemName*(list2.get(j)) + " " + sum);  
 }  
  
 duplicates.clear();  
 }  
  
 String [] whatever = new String[list2.size()];  
 list2.toArray(whatever);  
 return whatever;  
}

далее сортируем по кол-ву продаж по убыванию

// method to sort string array descending by numbers after whitespace  
static String[] bubbleSort(String[] inputStr3) {  
 List<String> list1 = new ArrayList<>();  
  
 for (int i = 0; i < inputStr3.length; i++){  
 list1.add(inputStr3[i]);  
 }  
 list1.add("!!!temp!!!");  
 int n = list1.size();  
  
 for (int i = 0; i < (n - 1); i++) {  
  
 for (int j = 1; j < (n - i - 1); j++) {  
  
 if (*amount*(list1.get(j - 1)) < (*amount*(list1.get(j)))) {  
 list1.set(n - 1, list1.get(j - 1));  
 list1.set(j - 1, list1.get(j));  
 list1.set(j, list1.get(n - 1));  
 }  
 }  
 }  
 list1.remove(n - 1);  
  
 String[] sortedStringArr = list1.toArray(new String[0]);  
 return sortedStringArr;  
}

Далее элементы с одинаковым количеством продаж сортируем

Лексикографически по возрастанию с заменой в основном массиве

Полный код:

public class Sales {  
  
 static String [] ShopOLAP(int N, String [] items){  
 String [] duplicatesRemoved = *removeDuplicates*(items);  
 String [] finalResult = *bubbleSort*(duplicatesRemoved);  
 //sorting same sales groups (same sales numbers after whitespace)  
 int offset = 0;  
 for (int i = 1; i < finalResult.length; i++){  
 if (*amount*(finalResult[i]) != *amount*(finalResult[i - 1])){  
 Arrays.*sort*(finalResult, offset, i);  
 offset = i;  
 }  
 }  
 return finalResult;  
 }  
  
 static String[] removeDuplicates(String[] inputString) {  
 int sum = 0;  
  
 List<String> list2 = new ArrayList<>();  
 for (int i = 0; i < inputString.length; i++){  
 list2.add(inputString[i]);  
 }  
  
 //creating array for duplicate item names  
 ArrayList<Integer> duplicates = new ArrayList<>();  
 int length = list2.size();  
  
 //outer loop for name check  
 for (int j = 0; j < length; j++) {  
 sum = *amount*(list2.get(j));  
  
  
 for (int i = j + 1; i < length; i++) {  
 //calculating sum of sales of duplicate names  
 if (*itemName*(list2.get(j)).equals(*itemName*(list2.get(i)))){  
 duplicates.add(i);  
 sum = sum + *amount*(list2.get(i));  
 }  
 }  
  
 if (duplicates.size() == 1) {  
 int a = duplicates.get(0);  
 list2.remove(a);  
 length--;  
 //replacing current item with same name + total sales  
 list2.set(j, *itemName*(list2.get(j)) + " " + sum);  
 }  
  
 if (duplicates.size() > 1) {  
 int dupSize = duplicates.size();  
 for (int r1 = dupSize - 1; r1 >= 0; r1--){  
 int a = duplicates.get(r1);  
 list2.remove(a);  
 }  
 length = length - dupSize;  
 //replacing current item with same name + total sales  
 list2.set(j, *itemName*(list2.get(j)) + " " + sum);  
 }  
  
 duplicates.clear();  
 }  
  
 String [] whatever = new String[list2.size()];  
 list2.toArray(whatever);  
 return whatever;  
 }  
  
 //method to extract numbers after whitespace as an int  
 static int amount (String inputSubstr){  
 int finalAmount = 0;  
 int spaceIndex = 0;  
 spaceIndex = inputSubstr.indexOf(' ');  
 String subs = inputSubstr.substring(spaceIndex + 1);  
 finalAmount = Integer.*parseInt*(subs);  
 return finalAmount;  
 }  
  
 //method to extract item name (substring before whitespace)  
 static String itemName (String inputStr1){  
 int spaceIndex1 = 0;  
 spaceIndex1 = inputStr1.indexOf(' ');  
 String subs = inputStr1.substring(0, spaceIndex1);  
 return subs;  
 }  
  
 // method to sort string array descending by numbers after whitespace  
 static String[] bubbleSort(String[] inputStr3) {  
 List<String> list1 = new ArrayList<>();  
  
 for (int i = 0; i < inputStr3.length; i++){  
 list1.add(inputStr3[i]);  
 }  
 list1.add("!!!temp!!!");  
 int n = list1.size();  
  
 for (int i = 0; i < (n - 1); i++) {  
  
 for (int j = 1; j < (n - i - 1); j++) {  
  
 if (*amount*(list1.get(j - 1)) < (*amount*(list1.get(j)))) {  
 list1.set(n - 1, list1.get(j - 1));  
 list1.set(j - 1, list1.get(j));  
 list1.set(j, list1.get(n - 1));  
 }  
 }  
 }  
 list1.remove(n - 1);  
  
 String[] sortedStringArr = list1.toArray(new String[0]);  
 return sortedStringArr;  
 }  
  
  
  
 public static void main(String[] args) {  
 String[] test = {"bag32 22", "bag33 22"};  
 int n = 8;  
  
 String[] ttt = (*ShopOLAP*(n, test));  
 System.*out*.println(Arrays.*toString*(ttt));  
  
  
  
  
  
 }  
}