МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Череповецкий государственный университет»

**Лабораторная работа № 4**

**«Коллекции»**

**Выполнил:**

студент гр. 1ИВТпб-01-31оп

Климов А.Г.  
**Проверил:** преподаватель

Пышницкий К.М.  
Отметка о зачете:

Череповец

2018 год

**Задание**

Написать и отладить три программы. Первая программа демонстрирует использование контейнерных классов для хранения встроенных типов данных.

Вторая программа демонстрирует использование контейнерных классов для хранения пользовательских типов данных.

Третья программа демонстрирует использование алгоритмов работы с коллекциями.

**В программе № 1** выполнить следующее:

1. Создать объект-контейнер в соответствии с вариантом задания (тип 1) и заполнить его данными, тип которых определяется вариантом задания.

2. Просмотреть контейнер.

3. Изменить контейнер, удалив из него одни элементы и заменив другие.

4. Просмотреть контейнер, используя для доступа к его элементам итераторы.

5. Создать второй контейнер этого же класса и заполнить его данными того же типа, что и первый контейнер.

6. Изменить первый контейнер, удалив из него n элементов после заданного и добавив затем в него все элементы из второго контейнера.

7. Просмотреть первый и второй контейнеры.

**В программе № 2** выполнить то же самое, но для данных пользовательского типа.

**В программе № 3** выполнить следующее:

1. Создать контейнер, содержащий объекты пользовательского типа. Тип контейнера выбирается в соответствии с вариантом задания (тип1).

2. Отсортировать его по возрастанию элементов.

3. Просмотреть контейнер.

4. Используя подходящий алгоритм, найти в контейнере элемент, удовлетворяющий заданному условию.

5. Переместить элементы, удовлетворяющие заданному условию в другой (предварительно пустой) контейнер. Тип второго контейнера определяется вариантом задания (тип2).

6. Просмотреть второй контейнер.

7. Отсортировать первый и второй контейнеры по убыванию элементов.

8. Просмотреть их.

9. Получить третий контейнер путем слияния первых двух. Тип третьего контейнера определить самостоятельно.

10. Просмотреть третий контейнер.

**Примечания:**

При перемещении элементов ассоциативного контейнера в неассоциативный перемещаются только данные (ключи не перемещаются). И, наоборот, при перемещении элементов не ассоциативного контейнера в ассоциативный должен быть сформирован ключ.

Варианты заданий

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Первый  контейнер( тип 1) | Второй  контейнер (тип 2) | Встроенный  тип данных |
| 5 | ArrayList | HashSet | char |

Варианты пользовательских классов

**5. БИБЛИОТЕКА**

имя – String

автор – String

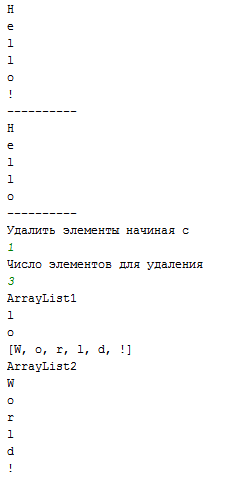
стоимость – float

**Текст программы 1**

**LR4\src\com.lr4.main\Main**

**package** com.lr4.main;  
  
**import** java.util.ArrayList;  
**import** java.util.Iterator;  
**import** java.util.Scanner;  
  
**public class** Main {  
 **public static void** main(String[] args) {  
 *//1. Создание ArrayList1 и заполнение* ArrayList al1 = **new** ArrayList();  
 String str = **"Hello!"**;  
 **for** (**char** c : str.toCharArray()) {  
 al1.add(c);  
 }  
  
 *//2. Просмотр контейнера* **for** (**int** i = 0; i < al1.size(); i++) {  
 System.***out***.println(al1.get(i));  
 }  
  
 System.***out***.println(**"----------"**);  
 *//3. Изменение контейнера* al1.set(0, **'H'**);  
 al1.remove(5);  
  
 *//4. Просмотр контейнера с помощью итератора* Iterator iter = al1.iterator();  
 **while**(iter.hasNext()){  
 System.***out***.println(iter.next());  
 }  
  
 *//5. Создание ArrayList2 и заполнение* ArrayList al2 = **new** ArrayList(5);  
 al2.add(**'W'**);  
 al2.add(**'o'**);  
 al2.add(**'r'**);  
 al2.add(**'l'**);  
 al2.add(**'d'**);  
 al2.add(**'!'**);  
  
 System.***out***.println(**"----------"**);  
 *//6. Удаление из первого контейнера n элементов после заданного  
 //и добавление в него всех элементов из второго контейнера* System.***out***.println(**"Удалить элементы начиная с "**);  
 Scanner in = **new** Scanner(System.***in***);  
 **int** n = in.nextInt();  
 n--;  
 System.***out***.println(**"Число элементов для удаления "**);  
 **int** m = in.nextInt();  
 m--;  
  
 **for** (**int** i = m + n; i >= n; i--) {  
 al1.remove(i);  
 }  
 al1.add(al2);  
  
 *//7. Вывод содержимого контейнеров* System.***out***.println(**"ArrayList1 "**);  
 **for** (**int** i = 0; i < al1.size(); i++) {  
 System.***out***.println(al1.get(i));  
 }  
  
 System.***out***.println(**"ArrayList2 "**);  
 **for** (**int** i = 0; i < al2.size(); i++) {  
 System.***out***.println(al2.get(i));  
 }  
 }  
}

**Тестирование 1**

****

**Текст программы 2**

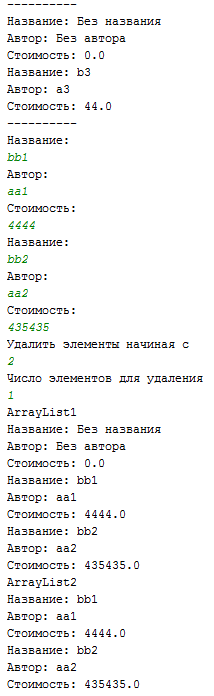
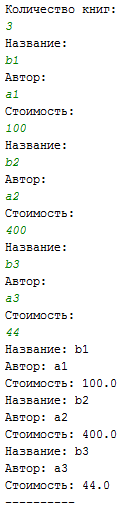
**LR4\_2\src\com.lr4.main\Main**

**package** com.lr4.main;  
  
**import** java.util.ArrayList;  
**import** java.util.Iterator;  
**import** java.util.Scanner;  
  
**public class** Main {  
 **public static void** main(String[] args) {  
  
 *//1. Создание ArrayList1 и заполнение* ArrayList<Library> al1 = **new** ArrayList<Library>();  
 System.***out***.println(**"Количество книг: "**);  
  
 Scanner inn = **new** Scanner(System.***in***);  
 **int** nn = inn.nextInt();  
  
 **for** (**int** i = 0; i <nn; i++) {  
 Scanner in = **new** Scanner(System.***in***);  
 System.***out***.println(**"Название: "**);  
 String nm = in.nextLine();  
 System.***out***.println(**"Автор: "**);  
 String a = in.nextLine();  
 System.***out***.println(**"Стоимость: "**);  
 **double** c = in.nextDouble();  
 al1.add(**new** Library(nm, a, c));  
 }  
  
 *//2. Просмотр контейнера* **for** (**int** i = 0; i < al1.size(); i++) {  
 al1.get(i).print();  
 }  
  
 *//3. Изменение контейнера* al1.remove(0);  
 Library tmp = **new** Library(**"Без названия"**, **"Без автора"**, 0);  
 al1.set(0, tmp);  
  
 System.***out***.println(**"----------"**);  
 *//4. Просмотр контейнера с помощью итератора* Iterator<Library> iter = al1.iterator();  
 **while**(iter.hasNext()){  
 iter.next().print();  
 }  
  
 System.***out***.println(**"----------"**);  
 *//5. Создание ArrayList2 и заполнение* ArrayList<Library> al2 = **new** ArrayList<Library>();  
 **for** (**int** i = 0; i <2; i++) {  
 Scanner in = **new** Scanner(System.***in***);  
 System.***out***.println(**"Название: "**);  
 String nm = in.nextLine();  
 System.***out***.println(**"Автор: "**);  
 String a = in.nextLine();  
 System.***out***.println(**"Стоимость: "**);  
 **double** c = in.nextDouble();  
 al2.add(**new** Library(nm, a, c));  
 }  
  
 *//6. Удаление из первого контейнера n элементов после заданного  
 //и добавление в него всех элементов из второго контейнера* System.***out***.println(**"Удалить элементы начиная с "**);  
 Scanner in = **new** Scanner(System.***in***);  
 **int** n = in.nextInt();  
 n--;  
 System.***out***.println(**"Число элементов для удаления "**);  
 **int** m = in.nextInt();  
 m--;  
  
 **for** (**int** i = m + n; i >= n; i--) {  
 al1.remove(i);  
 }  
 al1.addAll(al2);  
  
 *//7. Вывод содержимого контейнеров* System.***out***.println(**"ArrayList1 "**);  
 **for** (**int** i = 0; i < al1.size(); i++) {  
 al1.get(i).print();  
 }  
  
 System.***out***.println(**"ArrayList2 "**);  
 **for** (**int** i = 0; i < al2.size(); i++) {  
 al2.get(i).print();  
 }  
 }  
}

**LR4\_2\src\com.lr4.main\Library**

**package** com.lr4.main;  
  
**public class** Library {  
 **private** String **name**;  
 **private** String **author**;  
 **private double cost**;  
  
 **public** Library(String iname, String iauthor, **double** icost)  
 {  
 **this**.**name** = iname;  
 **this**.**author** = iauthor;  
 **this**.**cost** = icost;  
 }  
  
 **public void** print(){  
 System.***out***.println(**"Название: "** + **this**.**name**);  
 System.***out***.println(**"Автор: "** + **this**.**author**);  
 System.***out***.println(**"Стоимость: "** + **this**.**cost**);  
 }  
}

**Тестирование 2**

****

**Текст программы 3**

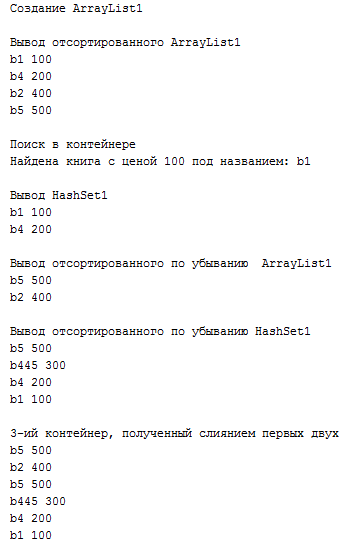
**LR4\_3\src\com.lr4.main\Main**

**package** com.lr4.main;  
  
**import** java.util.\*;  
  
**public class** Main {  
 **public static void** main(String[] args) {  
 **int** counter = 0;  
  
 System.***out***.println(**"Создание ArrayList1"**);  
 *//1. Создание ArrayList1 и заполнение* ArrayList<Library> al1 = **new** ArrayList<Library>();  
  
 al1.add(**new** Library(counter, **"b5"**, **"sdfsf"**, 500));  
 counter++;  
 al1.add(**new** Library(counter, **"b4"**, **"gff"**, 200));  
 counter++;  
 al1.add(**new** Library(counter, **"b2"**, **"34535"**, 400));  
 counter++;  
 al1.add(**new** Library(counter, **"b1"**, **"sdf"**, 100));  
 counter++;  
  
  
 *//2. Сортировка контейнера* Collections.*sort*(al1);  
  
 System.***out***.println(**"\nВывод отсортированного ArrayList1"**);  
 *//3. Вывод 1-го контейнера* **for**(**int** i =0 ; i<al1.toArray().**length**; i++)  
 {  
 System.***out***.println(al1.get(i).get\_name()+**" "**+al1.get(i).get\_cost());  
 }  
  
 System.***out***.println(**"\nПоиск в контейнере"**);  
 *//4. Используя подходящий алгоритм, найти в контейнере  
 //элемент, удовлетворяющий заданному условию* **for** (**int** i = 0; i < al1.size(); i++) {  
 **if** (al1.get(i).get\_cost() == 100){System.***out***.println(**"Найдена книга с ценой 100 под названием: "** + al1.get(i).get\_name());}  
 }  
  
 *//5. Переместить элементы, удовлетворяющие заданному  
 //условию в другой (предварительно пустой) контейнер* HashSet<Library> hs1 = **new** HashSet<Library>();  
 **for** (**int** i = 0; i < al1.size(); i++) {  
 **if** (al1.get(i).get\_cost() > 50 && al1.get(i).get\_cost() < 300)  
 {hs1.add(al1.get(i)); al1.remove(i); i--;}  
 }  
  
 System.***out***.println(**"\nВывод HashSet1 "**);  
 *//6. Вывод 2-го контейнера* **for** (Iterator<Library> it = hs1.iterator(); it.hasNext(); ) {  
 Library f = it.next();  
 System.***out***.println(f.get\_name() + **" "** + f.get\_cost());  
 }  
  
 *//Iterator<Library> iter = hs1.iterator();  
 //while(iter.hasNext()) {  
 // System.out.println(iter.next());  
 //}  
  
 //7. Сортировка контейнеров по убыванию  
 //Collections.reverse(al1);* Collections.*sort*(al1, Collections.*reverseOrder*());  
  
 ArrayList<Library> list = **new** ArrayList<Library>(hs1);  
 list.add(**new** Library(counter, **"b5"**, **"sdfsf"**, 500));  
 list.add(**new** Library(counter, **"b445"**, **"sdfsf"**, 300));  
 *//Collections.sort(list, Collections.reverseOrder());  
 //Collections.reverse(list);* Collections.*sort*(list, Collections.*reverseOrder*());  
 hs1 = **new** LinkedHashSet(list);  
  
 System.***out***.println(**"\nВывод отсортированного по убыванию ArrayList1 "**);  
 *//8. Вывод контейнеров* **for** (Iterator<Library> it = al1.iterator(); it.hasNext(); ) {  
 Library f = it.next();  
 System.***out***.println(f.get\_name() + **" "** + f.get\_cost());  
 }  
  
 System.***out***.println(**"\nВывод отсортированного по убыванию HashSet1 "**);  
 **for** (Iterator<Library> it = hs1.iterator(); it.hasNext(); ) {  
 Library f = it.next();  
 System.***out***.println(f.get\_name() + **" "** + f.get\_cost());  
 }  
  
 *//9. 3-ий контейнер, полученный слиянием первых двух* LinkedList ll1 = **new** LinkedList();  
 ll1.addAll(al1);  
 ll1.addAll(hs1);  
  
 System.***out***.println(**"\n3-ий контейнер, полученный слиянием первых двух "**);  
 *//10. Вывод 3-го контейнера* **for** (Iterator<Library> it = ll1.iterator(); it.hasNext(); ) {  
 Library f = it.next();  
 System.***out***.println(f.get\_name() + **" "** + f.get\_cost());  
 }  
 }  
}

**LR4\_3\src\com.lr4.main\Library**

**package** com.lr4.main;  
  
**import** java.io.Serializable;  
**import** java.util.Comparator;  
  
**public class** Library **implements** Comparable {  
 *//public class Library implements Comparable<Library>, Serializable {* **private int id**;  
 **private** String **name**;  
 **private** String **author**;  
 **private int cost**;  
  
 **public** Library(**int** iid, String iname, String iauthor, **int** icost)  
 {  
 **this**.**id** = iid;  
 **this**.**name** = iname;  
 **this**.**author** = iauthor;  
 **this**.**cost** = icost;  
 }  
  
 **public void** print(){  
 System.***out***.println(**"id: "** + **this**.**name**);  
 System.***out***.println(**"Название: "** + **this**.**name**);  
 System.***out***.println(**"Автор: "** + **this**.**author**);  
 System.***out***.println(**"Стоимость: "** + **this**.**cost**);  
 }  
  
 **public int** get\_id(){  
 **return this**.**id**;  
 }  
  
 **public int** get\_cost(){  
 **return this**.**cost**;  
 }  
  
  
 **public** String get\_name(){  
 **return this**.**name**;  
 }  
  
 */\* Перегрузка метода compareTo \*/* **public int** compareTo(Object obj)  
 {  
 Library tmp = (Library)obj;  
 **if**(**this**.**cost** < tmp.**cost**)  
 {  
 */\* текущее меньше полученного \*/* **return** -1;  
 }  
 **else if**(**this**.**cost** > tmp.**cost**)  
 {  
 */\* текущее больше полученного \*/* **return** 1;  
 }  
 */\* текущее равно полученному \*/* **return** 0;  
 }  
  
 *//@Override  
 //public int compareTo(Library o) {  
 // return this.name.compareTo(o.name);  
 //}*}

**Тестирование 3**

****