Introduction of bulk operations in ArrayList<>

Bulk operations are as follows

|  |  |
| --- | --- |
| **Method** | **Method Description** |
| **boolean** containsAll(Collection<?> c); | containAll returns true if this list contains all of element in specified collection |
| **boolean** addAll(Collection<? **extends** E> c); | addAll Appends all elements at the end of this list |
| **boolean** addAll(**int** index, Collection<? **extends** E> c); | addAll inserts all element at specified index |
| **boolean** removeAll(Collection<?> c); | removeAll removes all element in this list mentioned in specified list |
| **boolean** retainAll(Collection<?> c); | retainAll returns common elements from both the lists. |
| **default** **void** sort(Comparator<? **super** E> c) | sort the list. (Java 8 default method) |
| **void** clear(); | clear sets all element as null. |

All above methods are in the following program. I have heavily commented the code so it is simple to understand.

**package** arraylist;

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** ArrayListBulkOperations {

**public** **static** **void** main(String[] args) {

List<Integer> mainList = **new** ArrayList<Integer>();

List<Integer> tempList = **new** ArrayList<Integer>();

List<Integer> nextTempList = **new** ArrayList<Integer>();

/\*\*

\* Populate 3 lists

\* \*/

*populateMainList*(mainList);

*populateTempList*(tempList);

*populateNextTempList*(nextTempList);

System.***out***.println("mainList " + mainList);

System.***out***.println("tempList " + tempList);

System.***out***.println("nextTempList " + nextTempList);

System.***out***.println();

*containsAll*(mainList, tempList);

System.***out***.println();

*containsAll*(mainList, nextTempList);

System.***out***.println();

*addAllLists*(mainList, tempList);

System.***out***.println();

*addAllAtIndex*(mainList, tempList);

System.***out***.println();

*removeAllLists*(mainList, tempList);

System.***out***.println();

*retainsAllLists*(mainList, nextTempList);

System.***out***.println();

*sort*(mainList);

System.***out***.println();

*clear*(mainList);

}

/\*\*

\* Demonstrates the usage of

\* boolean containsAll(Collection<?> c);

\* \*/

**public** **static** **void** containsAll(List<Integer> mainList, List<Integer> tempList) {

**boolean** containsTempList = mainList.containsAll(tempList);

// [5, 7, 3, 1, 9] containsAll [7, 3, 9] true

// [5, 7, 3, 1, 9] containsAll [5, 3, 9, 44, 55] false

*display*(mainList, tempList, containsTempList);

}

/\*\*

\* Demonstrate the usage of

\* boolean addAll(Collection<? extends E> c);

\* \*/

**public** **static** **void** addAllLists(List<Integer> mainList, List<Integer> tempList) {

List<Integer> addAllList = **new** ArrayList<Integer>();

addAllList.addAll(mainList);

addAllList.addAll(tempList);

System.***out***.print("addAll(collection) ");

// addAll(collection) [5, 7, 3, 1, 9, 7, 3, 9]

*display*(addAllList);

System.***out***.println();

}

/\*\*

\* Demonstrate the usage of

\* boolean addAll(int index, Collection<? extends E> c);

\* \*/

**public** **static** **void** addAllAtIndex(List<Integer> mainList, List<Integer> tempList) {

List<Integer> addAllAtIndex = **new** ArrayList<Integer>();

addAllAtIndex.addAll(mainList);

addAllAtIndex.addAll(2, tempList);

System.***out***.print("addAll(index,collection) demo ");

// addAll(index,collection) demo [5, 7, 7, 3, 9, 3, 1, 9]

*display*(addAllAtIndex);

System.***out***.println();

}

/\*\*

\* Demonstrate the usage of

\* boolean removeAll(Collection<?> c);

\* \*/

**public** **static** **void** removeAllLists(List<Integer> mainList, List<Integer> tempList) {

List<Integer> removeAllList = **new** ArrayList<Integer>();

removeAllList.addAll(mainList);

System.***out***.print("List has " + removeAllList);

removeAllList.removeAll(tempList);

// List has [5, 7, 3, 1, 9] After removeAll [5, 1]

System.***out***.println(" After removeAll " + removeAllList);

}

/\*\*

\* Demonstrate the usage of

\* boolean retainAll(Collection<?> c);

\* \*/

**public** **static** **void** retainsAllLists(List<Integer> mainList, List<Integer> nextTempList) {

List<Integer> retainAllList = **new** ArrayList<Integer>();

// takes the common elements from the specified collection.

// intersection of 2 lists.

retainAllList.addAll(mainList);

// [5, 7, 3, 1, 9] [5, 3, 9, 44, 55]

System.***out***.println(mainList + " " + nextTempList);

// true

System.***out***.println(retainAllList.retainAll(nextTempList));

// [5, 3, 9]

System.***out***.println(retainAllList);

}

/\*\*

\* sort() method is default implementation of

\* List<> Interface added in JAVA 8.

\* \*/

**public** **static** **void** sort(List<Integer> mainList) {

mainList.sort(**null**);

// [1, 3, 5, 7, 9]

System.***out***.println(mainList);

}

/\*\*

\* Demonstrate the usage of

\* void clear()

\* All elements in list are made null.

\* \*/

**public** **static** **void** clear(List<Integer> mainList) {

mainList.clear();

// []

System.***out***.println(mainList);

}

**public** **static** **void** display(List<Integer> list1) {

System.***out***.print(list1);

}

**public** **static** **void** display(List<Integer> list1, List<Integer> list2, **boolean** output) {

System.***out***.println(list1 + " containsAll " + list2 + " " + output);

}

//Populate the list.

**public** **static** **void** populateMainList(List<Integer> mainList) {

mainList.add(5);

mainList.add(7);

mainList.add(3);

mainList.add(1);

mainList.add(9);

}

//Populate the list.

**public** **static** **void** populateTempList(List<Integer> tempList) {

tempList.add(7);

tempList.add(3);

tempList.add(9);

}

//Populate the list.

**public** **static** **void** populateNextTempList(List<Integer> nextTempList) {

nextTempList.add(5);

nextTempList.add(3);

nextTempList.add(9);

nextTempList.add(44);

nextTempList.add(55);

}

}

Output:

mainList [5, 7, 3, 1, 9]

tempList [7, 3, 9]

nextTempList [5, 3, 9, 44, 55]

[5, 7, 3, 1, 9] containsAll [7, 3, 9] true

[5, 7, 3, 1, 9] containsAll [5, 3, 9, 44, 55] false

addAll(collection) [5, 7, 3, 1, 9, 7, 3, 9]

addAll(index,collection) demo [5, 7, 7, 3, 9, 3, 1, 9]

List has [5, 7, 3, 1, 9] After removeAll [5, 1]

[5, 7, 3, 1, 9] [5, 3, 9, 44, 55]

true

[5, 3, 9]

[1, 3, 5, 7, 9]

[]