Find out the repeated elements in Int array

**public** **class** Exampleshow

{

@SuppressWarnings("unchecked")

**void** printRepeating(**int** arr[], **int** size)

{

@SuppressWarnings("rawtypes")

Map map=**new** HashMap();

**int** i;

System.***out***.println("Repeated elements are : ");

**for** (i = 0; i < size; i++)

{

**int** count=1;

**for**(**int** j=i+1; j<size; j++){

**if**(arr[i]==arr[j]){

count++;

}

}

**if**(!(map.containsKey(arr[i]))){

map.put(arr[i], count);

}

}

System.***out***.println(map+ “Order of n\*n only, O(n\*n), since two form loops only”);

}

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

{

Exampleshow repeat = **new** Exampleshow();

**int** arr[] = {4, 2, 4, 5,4,5,6,7,8,4,5,6,9};

**int** arr\_size = arr.length;

repeat.printRepeating(arr, arr\_size);

repeat.printRepeating1(arr, arr\_size);

}

**private** **void** printRepeating1(**int**[] arr, **int** size1) {

@SuppressWarnings("rawtypes")

Map map=**new** HashMap();

**int** i;

System.***out***.println("Repeated elements are : ");

**for** (i = 0; i < size1; i++)

{

**int** count=1;

**if**(!(map.containsKey(arr[i]))) {

map.put(arr[i], count);

} **else** {

**int** g=(((Integer)map.get(arr[i])).intValue());

g++;

map.put(arr[i], g);

}

}

System.***out***.println(map+”Order of n only, O(n)”);

}

}

Compare ArrayList and LinkedList in terms of time complexity and space complexity

|  |  |  |  |
| --- | --- | --- | --- |
| Type of DS | arrayList | LinkedList | reason |
| Insertion | O(n) | O(1) | Reason is same as explained for remove. |
| Deletion | O(n) | O(1) | Reason: LinkedList’s each element maintains two pointers (addresses) which points to the both neighbor elements in the list. Hence removal only requires change in the pointer location in the two neighbor nodes (elements) of the node which is going to be removed. While In ArrayList all the elements need to be shifted to fill out the space created by removed element. |
| Search | O(1) | O(n). | Reason: ArrayList maintains index based system for its elements as it uses array data structure implicitly which makes it faster for searching an element in the list. On the other side LinkedList implements **doubly linked list** which requires the traversal through all the elements for searching an element. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |