# HashMap + ArrayList Implementation Source-Code

import java.util.ArrayList;  
import java.util.HashMap;  
import java.util.Random;  
  
*/\*\*  
 \* Created by Stephen A. and Semida A.  
 \*/*public class HatHashMapArrayListImpl<T> implements Hat<T> {  
  
 private ArrayList<T> arrayObjects;   
 private HashMap<T,Integer> integerHashMap;   
  
 public HatHashMapArrayListImpl(){  
 arrayObjects = new ArrayList<>();  
 integerHashMap = new HashMap<>();  
 }  
@Override  
 public boolean isEmpty() {  
 return arrayObjects.isEmpty();  
 }  
@Override  
 public int size() {  
 return arrayObjects.size();  
 }  
@Override  
 public void give(T item) {  
 if (integerHashMap.get(item) != null)   
 return; // Returns  
  
 int s = arrayObjects.size(); // Object index   
 arrayObjects.add(item); // Add Object to arrayObjects  
 integerHashMap.put(item,s); // Add Object and Index   
  
 }  
  
@Override  
 public T take() {  
 int index = getRandomIndex(); // Get random index  
 T targetObject = arrayObjects.get(index); // Gets random element from arrayObjects  
  
 arrayObjects.remove(targetObject); // Remove object from arrayObjects  
 integerHashMap.remove(targetObject); // Remove index from hashMap  
  
 return targetObject; // Returns the Object to be removed.  
  
 }  
  
 private int getRandomIndex(){  
 Random r = new Random(); // Create Random Object.   
 return r.nextInt(arrayObjects.size()); // Returns random index   
 }  
  
 @Override  
 public String toString() {  
 return arrayObjects.toString(); // Returns the collection of objects.  
 }  
}

# Hat Interface Source-code

*/\*\*  
 \* Created by Stephen A.  
 \*/*public interface Hat<T> {  
  
 */\*\*  
 \** ***@return*** *Returns <b>true</b> if empty, <b>false</b> if more than 0 elements  
 \*/* boolean isEmpty();  
  
 */\*\*  
 \** ***@return*** *Returns size the size  
 \*/* int size();  
  
 */\*\*  
 \** ***@param*** *item The item to place into the collection  
 \*/* void give(T item);  
  
  
 */\*\*  
 \** ***@return*** *Returns a removes and returns a random element from the collection  
 \*/* T take();  
}