CS 314 Final Review — Hash Table Contains – **Solution**

Hash Tables

Write an instance method for the provided `HashTable314` class which will determine whether a given element is present in the table. This hash table implementation uses linked list buckets to handle hash collisions. These linked lists are **not** kept in sorted order. Elements in the hash table are placed in the bucket corresponding to their hash code modulo the length of the array.

This problem is about Hash Tables which use buckets for collisions. If you would like practice with a Hash Table which uses linear probing, revisit the extra section problem from Nov. 25

```
/* Pre: val != null
 * Post: returns true iff val is contained in the hash table.
 *       The hash table is unchanged as a result of this operation.
 */
 public boolean contains(E val);
```

You may use the following `HashTable314` implementation.

```
public class HashTable314<E> {
    private static final int INITIAL_CAPACITY = 10;

    private BucketNode<E>[] con;
    private int size;

    private static class BucketNode<E> {
      private E data;
      private BucketNode<E> next;
    }
}
```

You may not use or assume any other methods exist in the `HashTable314` class. Do not use any other Java classes or methods.

```
/* Pre: val != null
* Post: returns true iff val is contained in the hash table.
*       The hash table is unchanged as a result of this operation.
*/
public boolean contains(E val) {
    int index = Math.abs(val.hashCode() % con.length);
    return bucketContains(val, con[index]);
}

// Used to determine if val is present in the linked list
public boolean bucketContains(E val, BucketNode<E> n){
    if(n == null)
        return false;
    return n.data.equals(val) || bucketContains(val, n.next);
}
```

This was a very simple problem to test fundamental knowledge about bucket Hash tables. Some red herrings were thrown in like `INITIAL_CAPACITY` and the `size` instance variable to make sure you calculated the index correctly. The bucketContains method in the solution is just a short way to write a contains method for a linked list recursively (although an iterative solution would have been just fine as well).