Hashtable

Problem Statement

You are tasked with implementing a simple hashtable in Java using the separate chaining method to handle collisions. The provided skeleton code includes an inner class `Entry` that represents key-value pairs, as well as the basic structure for the hashtable itself. Your goal is to complete the implementation and further enhance it with additional features.

Tasks:

1. Implement the Constructor: Complete the constructor `MyHashtable(int capacity)` to initialize the hashtable with a specified capacity.
2. Implement `**put()**` Method: Write the `put(Object key, Object value)` method to insert a key-value pair into the hashtable. Handle collisions using the separate chaining method, where each entry in the hashtable can link to a chain of entries.
3. Implement `**get()**` Method: Write the `get(Object key)` method to retrieve the value associated with a given key. Ensure the method efficiently traverses the chain if necessary.
4. Implement `**remove()**` Method: Write the `remove(Object key)` method to remove the key-value pair associated with a given key. Ensure that the chain is properly maintained after removal.

Additional Tasks:

1. Use Generics: Modify the implementation to use generics so that the hashtable can store key-value pairs of any type.
2. Handle Resizing: Implement a method to resize the hashtable dynamically when the load factor exceeds a certain threshold, rehashing all existing keys.
3. Implement `containsKey()` Method: Add a method `containsKey(K key)` to check if a given key exists in the hashtable.
4. Implement `size()` and `isEmpty()` Methods: Implement methods to return the number of key-value pairs in the hashtable (`size()`) and to check if the hashtable is empty (`isEmpty()`).
5. Implement `clear()` Method: Add a method `clear()` to remove all entries from the hashtable.
6. Iterate through the Hashtable: Implement an iterator that allows traversal of all key-value pairs in the hashtable.