Situation

The software development company you work for has recently been awarded a contract to create an accounts receivable management system for a local law firm. As a member of the development team, you have been tasked to write the engine that will drive the system**. You've decided to write a prototype application to test the feasibility of using an implementation of the Map ADT as the foundation for the engine.**

You can use a simple [AccountRecordhttps://slcc.instructure.com/images/popout.png](https://slcc.instructure.com/files/28767972/download?verifier=3qHo1g8ZOu4B1zptmjEOA8SPkSOY4KtRqNWRDTsr) class you remember from your days at SLCC in your prototype (an account record provided by the law firm will be used in the delivered product). **The account number and an instance of the AccountRecord will serve nicely as the key-value pair to be stored in the map. You also determine to take advantage of the memory management, searching, and balancing features of an AVL tree as the backing store for your map.**

If all goes well and your concept proves viable, you'll be looking at a quick promotion and a really big raise from your boss.

Specification

1. Create class AVLMap that implements Gray's Map interface and uses the implementation of Gray's AVLTree as the backing store.
2. Use class Entry<K, V> from Gray's Map package to store key-value pairs in the AVL tree. K and V will represent the account number and account record respectively.
3. Order the entries in the AVL tree by the key (account number) in the key-value entry.
4. Create a test application to validate your prototype that allows for adding, searching, removing and editing account records.

Admin

1. Grading
   * 0 points if your program does not compile.
   * +5 for comments, indentation and placement of {} per [Style Guide](http://www.cs.slcc.edu/style-guide.shtml).
   * +5 for each specification met.
2. Submission: Attach an executable JAR file that also contains your .java source code files.

K is the account number

V is the account record

May need to re-insert, right after Constructor

//override methods from map interface  
 44 public void clear()  
 45 {  
 46 map.clear();  
 47 }  
 48   
 49 public boolean containsKey(K key)  
 50 {  
 51 return false;  
 52 }  
 53   
 54 public boolean containsValue(V value)  
 55 {  
 56 return false;  
 57 }  
 58   
 59 public V getKey(K key)  
 60 {  
 61   
 62 }  
 63   
 64 public boolean isEmpty()  
 65 {  
 66 return this.size() == 0;  
 67 }  
 68   
 69 public V put( K key, V value )  
 70 {  
 71 return value;  
 72 }   
 73 public V remove( K key )  
 74 {  
 75   
 76 }  
 77   
 78 public int size()  
 79 {  
 80 return map.size();  
 81 }