

Homework #4

You may assume that you always map strings to integers, and therefore you need not worry about using generics.

Here are the methods that your map must implement.

void put (**String** key, **Integer** value);

insert this key-value pair into the map under the following rules:

If *key* already exists in the map, change its value to *value*, overwriting what was there.

If *key* is not already there, insert this key-value pair into the map.

boolean containsKey(**String** key);

returns true iff *key* already exists in the map.

boolean containsValue(**Integer** value);

returns true iff value appears in the map. (N.B. Since the values need not be unique, this doesn't necessarily tell you anything interesting.)

boolean remove (**Object** key);

If *key* is in the map, remove it. (This is a little bit tricky).

Integer get(**String** key);

If *key* is in the map, return the associated value.

If *key* is **NOT** in the map, return null.

Set<Pair> entrySet();

Returns a Set of all pairs in the map.

Collection<Integer> values();

Returns a list of all of the associated values in the map.

Set<String> keySet();

Returns a set of all keys in the map.

int size();
returns the number of elements in the map

boolean isEmpty();
returns true iff the number of elements in the map is 0.

You will use a hash table as the underlying structure.

The declaration for the private data member should be:

```
private ArrayList<LinkedList<Pair>> theTable;
```

Pair should be an inner static class with the following methods.

```
Pair(String str, Integer value);  
    Constructs a pair  
String getKey();  
    Returns the key  
Integer getValue();  
    Returns the value  
Integer setValue(Integer value);  
    Changes the value to a new value and returns the old value.
```

As always, your code must be heavily commented and documented.