The Java interface `College` is implemented by objects that store information about colleges and universities. The interface declares methods for accessing a `College` object's name, tuition, and the region in which it is located. The interface also specifies constants for naming the regions.

```java
public interface College {
    public final static String NORTHEAST = "Northeast";
    public final static String SOUTHEAST = "Southeast";
    public final static String NORTHWEST = "Northwest";
    public final static String MIDWEST   = "Midwest";
    public final static String SOUTHWEST = "Southwest";
    public final static String WEST      = "West";
    public final static String SOUTH     = "South";

    public String getName();   // returns name of college
    public String getRegion(); // returns region of college
    public int getTuition();   // returns tuition for college
}
```

The class `CollegeGroup` stores information about a group of colleges/universities. Part of the `CollegeGroup` class declaration is shown below.

```java
import java.util.ArrayList;

public class CollegeGroup {
    private College[] myColleges;  // myColleges.length is # colleges

    // precondition:  there exists a College in this group
    //                whose name is collegeName, call this
    //                myColleges[index]
    // postcondition: myColleges[index].getTuition() == newTuition, i.e.,
    //                the College with collegeName has its
    //                newTuition as its tuition

    public void updateTuition(String collegeName,
                              int newTuition)
    {
        // you will write this code
    }
}
```
// precondition:  low <= high
// postcondition: returns ArrayList of College objects
// from this group in specified region
// whose tuition is between (including)
// low and high, i.e., low <= tuition <= high

public ArrayList getCollegeList(String region, int low, int high) {
    // you will write this code
}

The following chart shows an example of colleges/universities that could appear in an object of type CollegeGroup.

<table>
<thead>
<tr>
<th></th>
<th>Name</th>
<th>Region</th>
<th>Tuition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Colgate University</td>
<td>Northeast</td>
<td>$27,025</td>
</tr>
<tr>
<td>1</td>
<td>Duke University</td>
<td>Southeast</td>
<td>$26,000</td>
</tr>
<tr>
<td>2</td>
<td>Kalamazoo College</td>
<td>Midwest</td>
<td>$19,764</td>
</tr>
<tr>
<td>3</td>
<td>Stanford University</td>
<td>West</td>
<td>$25,917</td>
</tr>
<tr>
<td>4</td>
<td>Florida International University</td>
<td>Southeast</td>
<td>$10,800</td>
</tr>
<tr>
<td>5</td>
<td>Dartmouth College</td>
<td>Northeast</td>
<td>$27,764</td>
</tr>
<tr>
<td>6</td>
<td>Spelman College</td>
<td>Southeast</td>
<td>$11,455</td>
</tr>
</tbody>
</table>

Part A

Write the CollegeGroup method updateTuition, which is described as follows. Method updateTuition associates a new tuition with the college whose name is passed as a parameter. Assume that the objects stored in the array myColleges are instances of a class CollegeImpl that implements the College interface and whose declaration is partially shown below.

```java
public class CollegeImpl implements College {
    private String myName;
    private String myRegion;
    private int myTuition;

    public CollegeImpl(String name, String region, int tuition) {
        myName = name;
        myRegion = region;
        myTuition = tuition;
    }

    // more methods not shown
}
```

Note that neither the College interface nor the CollegeImpl class specify a method for changing tuition; i.e., CollegeImpl is an immutable class.
In writing `updateTuition` you should create a new `CollegeImpl` object for the specified college, with the same name and region, but a new tuition. This new object should be stored in the same location of `myColleges` as the original object.

Complete method `updateTuition` below.

class CollegeGroup
{
   //not all methods, fields shown
   // precondition:  there exists a College in this group
   // whose name is collegeName, call this
   // myColleges[index]
   // postcondition: myColleges[index].getTuition() == newTuition, i.e.,
   // the College with collegeName has
   // newTuition as its tuition
   public void updateTuition(String collegeName, int newTuition)
   {

   }
}

Part B

The table below is repeated for your convenience.

<table>
<thead>
<tr>
<th></th>
<th>Name</th>
<th>Region</th>
<th>Tuition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Colgate University</td>
<td>Northeast</td>
<td>$27,025</td>
</tr>
<tr>
<td>1</td>
<td>Duke University</td>
<td>Southeast</td>
<td>$26,000</td>
</tr>
<tr>
<td>2</td>
<td>Kalamazoo College</td>
<td>Midwest</td>
<td>$19,764</td>
</tr>
<tr>
<td>3</td>
<td>Stanford University</td>
<td>West</td>
<td>$25,917</td>
</tr>
<tr>
<td>4</td>
<td>Florida International University</td>
<td>Southeast</td>
<td>$10,800</td>
</tr>
<tr>
<td>5</td>
<td>Dartmouth College</td>
<td>Northeast</td>
<td>$27,764</td>
</tr>
<tr>
<td>6</td>
<td>Spelman College</td>
<td>Southeast</td>
<td>$11,455</td>
</tr>
</tbody>
</table>

Write the `CollegeGroup` method `getCollegeList`, which is described as follows. Method `getCollegeList` returns an ArrayList of colleges that are located in the specified region and whose tuition is in the range between `low` and `high`, inclusive. The size of the ArrayList should be equal to the number of colleges that
meet the criteria of region and tuition range.

For example, if the object `colleges` is an instance of the class `CollegeGroup` and represents the entries shown in the chart above, the call

```java
ArrayList list = colleges.getCollegeList(College.SOUTHEAST, 10000, 20000);
```

should store in `list` an ArrayList of two elements containing objects representing Florida International University and Spelman College (note that Duke University is not included because its tuition is not in the specified range and Kalamazoo College is not included because it is not in the specified region).

Complete the method below.

```java
public class CollegeGroup
{
    // not all methods, fields shown

    private College[] myColleges; // myColleges.length is # colleges

    // precondition:  low <= high
    // postcondition: returns ArrayList of College objects
    //                from this group in specified region
    //                whose tuition is between (including)
    //                low and high, i.e., low <= tuition <= high

    public ArrayList getCollegeList(String region, int low, int high)
    {
        // Implementation goes here
    }
}
```

---

*Owen L. Astrachan*