CSCI 3357: Database System Implementation
Homework Assignment 2
Due Thursday, September 14

Make the following two modifications to the SimpleDB class `Page`.

1. The methods `setInt`, `setBytes`, and `setString` do not check that the specified value will fit into the byte buffer at the specified offset. If it doesn't, the `ByteBuffer` class will throw an `IndexOutOfBounds` exception. Modify these methods so that they write the specified value only if it fits. If the value does not fit, the methods should print a descriptive message and ignore the request. For example, if the capacity of the page's byte buffer is 400 bytes, then the call `p.setInt(398,12)` should print the following message:

```
ERROR: The integer 12 does not fit at location 398 of the page
```

Note: To get the capacity of a `ByteBuffer` object, call its `capacity()` method.

2. Currently, the `setString` method writes a string as a "blob" of bytes, prepended by an integer denoting the length of the blob. Another way to implement the method is to write each individual character of the string to the byte buffer, followed by the delimiter character '\0'. Modify the method `setString` to use this strategy. Modify the method `getString` analogously.

Your implementation should use the `ByteBuffer` methods `getChar` and `putChar` to read and write the characters to the byte buffer. If you do not understand how to use these methods, look up their JavaDoc documentation. These methods encode each character using two bytes, regardless of the specified charset. Thus you will also need to modify the page's `maxLength` method. In doing so, you should discover that the method no longer needs to reference CHARSET at all.

Submit your revised file *Page.java* to Canvas. Please do not change its class name or its package, as I will be grading your file by inserting it directly into my SimpleDB environment.

Also, please consider the following suggestions:

A. Before doing this assignment (or any assignment that changes SimpleDB), you should create a backup copy of the SimpleDB code. That way, if you wind up being totally unable to debug your code, you can always restore back to a working version.

B. One way to test your revised code is to run some SimpleDB clients and see if they still work. As a general rule, this is a terrible idea. The SimpleDB engine does a lot of things behind the scenes that you are not yet aware of, and your code may cause it to fail in ways that you won't be able to understand. Much better is to create and run a test program. The file *FileTest.java* in the package `simpledb.file` is a simple example of a test program. I wrote a test program specifically for this assignment, called *HW2Test.java*, which I plan to use when grading your code. Feel free to download it for your own purposes.

C. Once you get your code to pass the test program, you should then try running the SimpleDB client programs. But be warned that your new version of SimpleDB will be incompatible with the two *studentdb* databases that you created for HW0. (You understand why, right?) So you will need to delete them and re-create with your new version.