CSC207H: Exercise 3

Due date: 10:00 a.m., Monday, November 3, 2008.

What to do for this exercise

1. Write a Java class "`Vector`" that obeys the specifications below. It must be in a package "`e3`".
2. Check the program into your Exercise 3 repository. As always, the thing we'll look for is a file called "`Vector.java`" in a directory "`e3`", and that directory may be either the "`exercises/e3`" itself or some descendant of it in the tree of directories.

Specifications for `Vector`

The `Vector` class models mathematical vectors, of the kind used to indicate distances with direction in multi-dimensional geometric spaces. A `Vector` object must have a private instance variable that is an array of `double`s containing the components of the mathematical vector. For example, if the mathematical vector is (1.5, 2.0, –4.6), and the `double[]` in the `Vector` object is called "`data`", then `data[0]` would contain `1.5`, `data[1]` would contain `2.0`, and `data[2]` would contain `-4.6`.

Your `Vector` class must have at least these features:

- A constructor that takes one parameter, an array of `double`s that will be copied into the `Vector`'s data array. It is important for you to copy the array element by element rather than simply copying the reference to the array.
- A `toString()` method that returns a string in just the right format: the elements of the array in order, each followed by a comma and a space. For our example vector, (1.5, 2.0, –4.6), the value returned by `toString()` must be "`1.5, 2.0, -4.6, `" (without the quotation marks). The number of decimal places shown should be the default number provided by Java when a `double` is converted to a `String` for printing. The comma and the space at the end of the `toString()` output look peculiar, but they make it easier to write the method. Notice that there is no newline character at the end.
- A `euclideanLength()` method that returns the usual square-root-of-the-sum-of-squares length of the mathematical vector.
- An exception class `E3exception` that is defined *inside* `Vector`. This exception must extend `RuntimeException`, and needs only two members: a no-argument constructor, and a one-argument constructor the argument of which is a `String`. Each constructor should simply call the parent class's constructor.
- An `add(other)` method that takes a `Vector` as parameter, and returns a new `Vector` the elements of which are made by adding the corresponding elements of this `Vector` and `other`. If this `Vector` and `other` do not have the same number of dimensions, an `E3exception` must be thrown.

- A `euclideanLength()` method that returns the usual square-root-of-the-sum-of-squares length of the mathematical vector.
- A `dotProduct(other)` methodthat takes a `Vector` as parameter, and returns a `double` that is the sum of the products of each element of this `Vector` multiplied by the corresponding element of `other`. For example, if the two vectors are (2, 4, 6) and (3, 5, 7), then the dot product is 6+20+42=68. If this `Vector` and `other` do not have the same number of dimensions, an `E3exception` must be thrown.

We will test your code by running JUnit tests, so there is no need for you to write a `main()` method, or worry about input or output. If you like, you are free to include those features, however.

Last change: $Date: 2008-10-23 15:51:13 -0400 (Thu, 23 Oct 2008) $