Creating Polyhedra (Java)

Thomas Kennedy

Contents:

Read the general instructions on turning in assignments.

1 The Problem

This assignment deals with a program takes three types of Polyhedra from an input file and constructs the appropriate objects.

In this exercise, you will be completing the Polyhedron Hierarchy- specifically the Composite class.

This is the same problem from Assignment 3. In this assignment, you will be solving a similiar problem, in Java.

1.1 Input

The program reads data from one file, *polyhedra1.txt*. File extensions on Linux may be arbitrary—i.e., this file could have been named with *.dat* as the extension.

```
sphere 1
cylinder 2 1
sphere 4
cylinder 2 3
composite 3
    sphere 3
    sphere 5
    sphere 7

composite 2
    cylinder 1 2
    sphere 5
sphere 3
```

Each polyhedron line is formatted as a *keyword*—i.e., the name of the polyhedron—and all appropriate attributes. A sphere is defined by a radius:

```
sphere 1
```

A cylinder with height 2 and radius 3 would take the form:

```
cylinder 2 3
```

A composite shape us defined by an integer representing the number, *n*, of polyhedra of which it is composed. It is then followed by *n* polyhedron input entries:

```
composite 2
    cylinder 1 2
    sphere 5
```

You may assume a valid input file. All input is well-formed.

1.2 Output

The output consists of two re dard output, one after the other.

1. A report listing all polyhe om polyhedra1.txt.
2. A report listing the result ra.

If the program is run with pol ut file, the following output should be generated:

```
Original Polyhedra
------------------------------------------------------------
[Sphere] (2.0, 2.0, 2.0)->Radius: 1.0 Diameter: 2.0
[Cylinder] (2.0, 2.0, 2.0)->Radius: 1.0 Height: 2.0
[Sphere] (8.0, 8.0, 8.0)->Radius: 4.0 Diameter: 8.0
[Cylinder] (6.0, 6.0, 2.0)->Radius: 3.0 Height: 2.0
[Composite] (14.0, 14.0, 14.0)->3 polyhedra
  [Sphere] (6.0, 6.0, 6.0)->Radius: 3.0 Diameter: 6.0
  [Sphere] (10.0, 10.0, 10.0)->Radius: 5.0 Diameter: 10.0
  [Sphere] (14.0, 14.0, 14.0)->Radius: 7.0 Diameter: 14.0

[Composite] (10.0, 10.0, 10.0)->2 polyhedra
  [Cylinder] (4.0, 4.0, 1.0)->Radius: 2.0 Height: 1.0
  [Sphere] (10.0, 10.0, 10.0)->Radius: 5.0 Diameter: 10.0

[Sphere] (6.0, 6.0, 6.0)->Radius: 3.0 Diameter: 6.0


Scaled Polyhedra (Clones)
------------------------------------------------------------
[Sphere] (4.0, 4.0, 4.0)->Radius: 2.0 Diameter: 4.0
[Cylinder] (4.0, 4.0, 4.0)->Radius: 2.0 Height: 4.0
[Sphere] (16.0, 16.0, 16.0)->Radius: 8.0 Diameter: 16.0
[Cylinder] (12.0, 12.0, 4.0)->Radius: 6.0 Height: 4.0
[Composite] (28.0, 28.0, 28.0)->3 polyhedra
  [Sphere] (12.0, 12.0, 12.0)->Radius: 6.0 Diameter: 12.0
  [Sphere] (20.0, 20.0, 20.0)->Radius: 10.0 Diameter: 20.0
  [Sphere] (28.0, 28.0, 28.0)->Radius: 14.0 Diameter: 28.0

[Composite] (20.0, 20.0, 20.0)->2 polyhedra
  [Cylinder] (8.0, 8.0, 2.0)->Radius: 4.0 Height: 2.0
  [Sphere] (20.0, 20.0, 20.0)->Radius: 10.0 Diameter: 20.0
```


The easiest way to generate the expected output is to run the sample executable solution I have provided. These two files are named as command-line parameters when the program is executed.

For example, if the sample data above is kept in polyhedra1.txt, to run this program, type:

```
java -jar CreatePolyhedra.jar polyhedra1.txt 2
```

or

```
make run
```

Where the latter command executes the `run` target in the makefile (which runs the first command for you). This allows us to use fewer keystrokes (which are expensive).

Run the compiled solution with both the provided input file and your own test input files.

Once you have completed your solution, compare the output generated by your solution to the output generated by my solution. The two sets must be identical.

(On a Windows system, you would omit the "./". If you are running from Eclipse or a similar development environment, you may need to review how to supply command-line parameters to a running program.)

1.3 Your Tasks

Complete `clone` (copy constructor portion), `scale`, and `tostring` functions for Composite class.

- Note that I have provided you a clone function and a skeleton for a copy constructor. You can either complete the copy constr copy constructor (updating `clone` as appropriate.
- Note the hints in the com v 8 Example 1.

2 Mechanics

2.1 Packages

For the sake of familiarity we will forgo proper packages (and leave all source code in the Default Package). Do **not** change/add any packages to this exercise. Do **not** add any package lines, e.g.,


```
package shapes;
```

If you change the package, you will fail automatically. Is this harsh, yes. However, I have explicitly made part of the assignment resisting the urge to use proper packages.

2.2 Grading

This is the same problem from Assignment 3. You are repeating the same exercise in Java. As such my grading criteria is stricter than it was in Assignment 1 & Assignment 2.

Tests 000 through 004 evaluate your program as a whole:

- Test 000 confirms that your code compiles and runs. This test discards all output.
- Test 001 evaluates how your program runs given input file and a scaling factor of 1. This test disregards output formatting.

- Test 002 evaluates how your program runs given input file and a scaling factor of 1. This test includes output formatting.
- Test 003 evaluates how your program runs given input file and a scaling factor greater than 1. This test disregards output formatting.
- Test 004 evaluates how your program runs given input file and a scaling factor greater than 1. This test includes output formatting.

2.3 Files

Files for this assignment appear in this directory or, if you are logged in to a CS Dept Linux server, in `~cs330/Assignments/polyhedra_java`.

2.4 Submitting

Files to Submit:

- Composite.java

Note that your submitted code must compile correctly—on our Linux servers—with the other code in that directory, using the compilation commands generated by the provided makefile. Do not alter any of the other source code files, nor change the Composite class interface in such a way that it can only be compiled with some other compiler or some other sequence of commands.

To submit your assignment, use the button below. You will receive a preliminary grade via email (to your ODU email account) and will also be able to check your grade from the course web page *Grades* button. This preliminary grade report will include any compilation errors encountered when compiling your code *(if you opted to submit non-compiling code)*.

Submit this assignment