CS 435/535 Assignment #7 – Spring 2021

Project Overview: In this project, you will use **Scheme** to complete the same shape project done previously. Again, there are four different kinds of shapes: `Sphere`, `Cylinder`, `Cone`, and `Cuboid`. All the shapes will be stored in a file. Each shape occupies one line and two attributes of a shape are separated by one or more spaces. The number of shapes in the file is unknown in advance. You can download from Blackboard a sample shape file named `shapes.dat` that contains the following shapes.

```
Cube#1 cuboid 1 1 1
Cube#2 cuboid 2 2 2
Cone#1 cone 1 1
Cyl#1 cylinder 1 1
Box#1 cuboid 2 4 6
Box#2 cuboid 10.5 21 10.5
UnitSphere sphere 1
LargeSphere sphere 100
Cone#2 cone 1 2
Cyl#2 cylinder 1 2
```

For this project, you need to create a Scheme file named `shapes.scm` to define a function named `perform` that will take a variable number of arguments. The first two arguments are required. The first argument specifies which action to be performed and it could be one of the following six actions: `count`, `print`, `min`, `max`, `total`, and `avg`. You can assume the caller will correctly supply one of these six actions. The second argument is the shape data file to perform the action on. If the data file can be opened for reading, you can assume the file will follow the correct format and contain correct attributes. If not, please print out an error message and return.

After the action and the data file, the caller may specify a list of test conditions. If this is the case, the action will be performed on the shapes that satisfy all the test conditions. Each test condition has three arguments in the `<name>` `<op>` `<value>` format. The `<name>` is a string, and it can be "type", "area" or "volume". The `<op>` is another string to represent one of the six relational operators (`"=="`, `"!="`, `">="`, `"<="`, `">"`, and `"<"`). The `<value>` is a string for "type" and a number for "area" or "volume". For example, "type" ">" "cyl", "area" "<=" 1000, and "volume" ">" 100.5 are three examples of test conditions. The number of arguments for optional test conditions must be a multiple of 3. If it is not a multiple of 3, please print out an error message and return. If it is a multiple of 3, you can assume the arguments supplied will follow the format described above. The following are three examples of function calls, with and without test conditions. Without a condition, the action will be performed on all the shapes in the file. With conditions, the action will be performed only on the shapes that satisfy all the conditions.

```
    (perform "min" "shapes.dat")
    (perform "max" "shapes.dat" "type" "==" "cuboid")
    (perform "print" "shapes.dat" "area" ">" 100 "area" "<" 200)
```

Although one function is required for this project, it is a good idea to write a lot of helper functions, and each helper function will achieve a part of the `perform` function. The sample execution at the end shows how your program will be loaded and tested. Please note the numeric values are not necessarily rounded to two decimal places.

What You Need To Do

- Create a directory named **`project7`** for this assignment. Download **`shapes.dat`** from Blackboard to the **`project7`** directory.
- Create a Scheme file named `shapes.scm` to define a function named `perform` to read the shapes file and to perform the requested action, with or without test conditions.

- When you are ready to submit your project, compress your **`project7`** directory into a single (compressed) zip file, **`project7.zip`**.
- Once you have a compressed zip file named **`project7.zip`**, submit that zip file to Blackboard.
- Your submission will be graded on `cs-parallel.ua.edu`. Make sure to test it on that machine before submission.
- Make sure to use `shapes.scm` for the file name and `perform` for the function name. Otherwise we may not be able to grade your submission.

Assignment #7 is due at 11:59pm on Monday, April 12. Late projects are not accepted.

A sample execution of the program

Assume a file named **`test.scm`** is created and it contains the following function calls.

```
(load "shapes.scm")
(perform "total" "xxxx.dat")
(perform "avg" "shapes.dat" 100 "<" "area" "<" 200)
(perform "print" "shapes.dat")
(perform "print" "shapes.dat" "type" "==" "cuboid" "area" ">=" 88)
(perform "count" "shapes.dat" "type" ">" "cyl")
(perform "count" "shapes.dat" "type" "!=" "cuboid")
```

The following are output after executing **`scheme --quiet < test.scm`**

```
Unable to open xxxx.dat for reading.

Incorrect number of arguments.

Cuboid: Cube#1, Length=1.00, Width=1.00, Height=1.00
        Surface Area: 6.00, Volume: 1.00
Cuboid: Cube#2, Length=2.00, Width=2.00, Height=2.00
        Surface Area: 24.00, Volume: 8.00
Cone: Cone#1, Radius=1.00, Height=1.00
        Surface Area: 7.58, Volume: 1.05
Cylinder: Cyl#1, Radius=1.00, Height=1.00
        Surface Area: 12.57, Volume: 3.14
Cuboid: Box#1, Length=2.00, Width=4.00, Height=6.00
        Surface Area: 88.00, Volume: 48.00
Cuboid: Box#2, Length=10.50, Width=21.00, Height=10.50
        Surface Area: 1102.50, Volume: 2315.25
Sphere: UnitSphere, Radius=1.00
        Surface Area: 12.57, Volume: 4.19
Sphere: LargeSphere, Radius=100.00
        Surface Area: 125663.71, Volume: 4188790.20
Cone: Cone#2, Radius=1.00, Height=2.00
        Surface Area: 10.17, Volume: 2.09
Cylinder: Cyl#2, Radius=1.00, Height=2.00
        Surface Area: 18.85, Volume: 6.28

Cuboid: Box#1, Length=2.00, Width=4.00, Height=6.00
        Surface Area: 88.00, Volume: 48.00
Cuboid: Box#2, Length=10.50, Width=21.00, Height=10.50
        Surface Area: 1102.50, Volume: 2315.25

There are 4 shapes.

There are 6 shapes.
```