CS2230 Computer Science II: Data Structures
Homework 2
Objects in Java: the ice cream shop

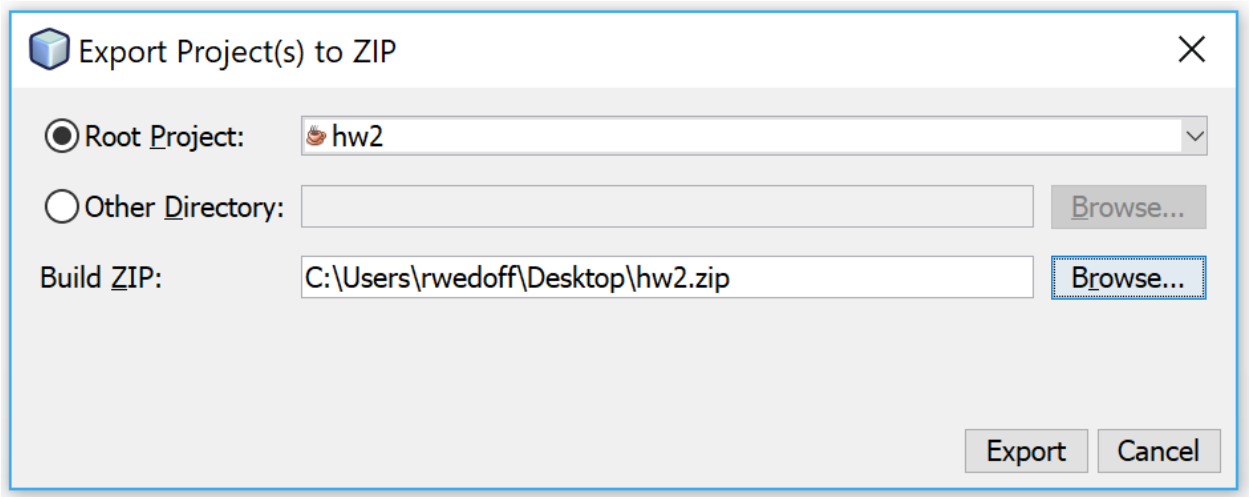
Due September 8th, 2017, 11:59pm

Goals for this assignment

- Write a program that uses objects
- Finish an application consisting of more than one Java class
- Use arrays of objects
- Write an algorithm using an array of objects

Submission Checklist

When you are done with this assignment, you should *export the project in NetBeans.*

1. Make sure you are exporting the right project
2. Open *File -> Export Project -> to Zip*
3. 

4. Export the root project of *hw2*
5. Under build ZIP, choose your file path, and name the ZIP file ***hw2.zip*****.**
   (Choose this wisely so you don't have to try and find it in a random location).
6. Upload *hw2.zip* to ICON under Assignments > Homework 2. Physical paper copies or images are not acceptable.

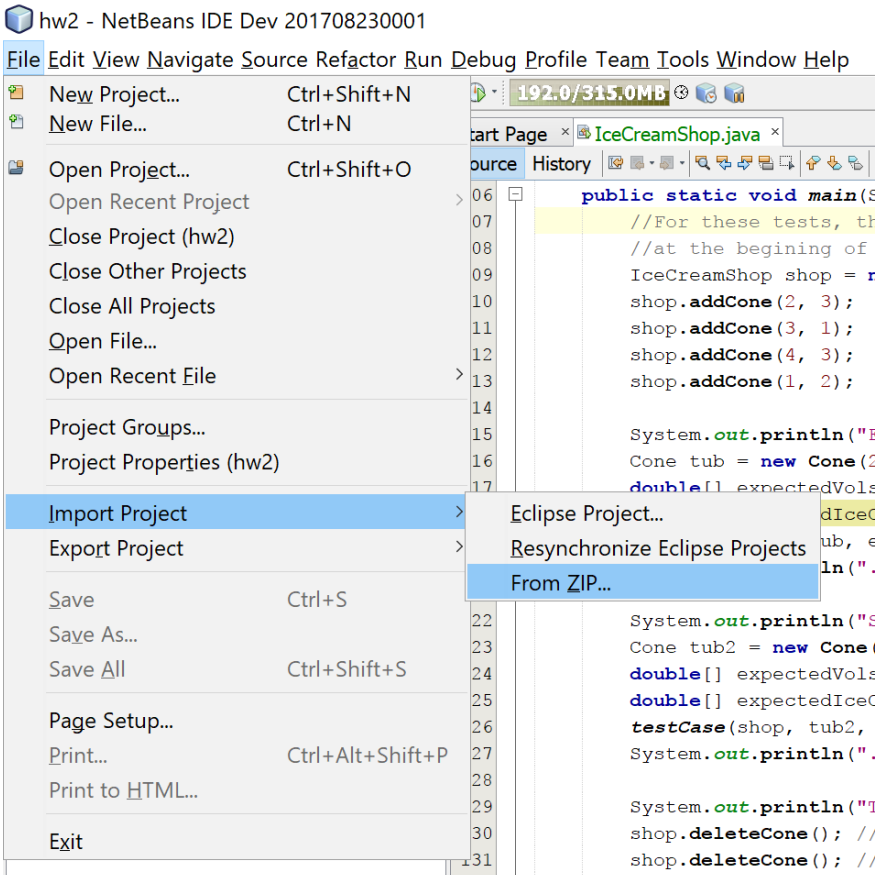
Things to check for:

- ✓ Does IceCreamShop.java run properly?
- ✓ Did you double check your ICON submission to be sure you uploaded the right zip file, containing the right java files?

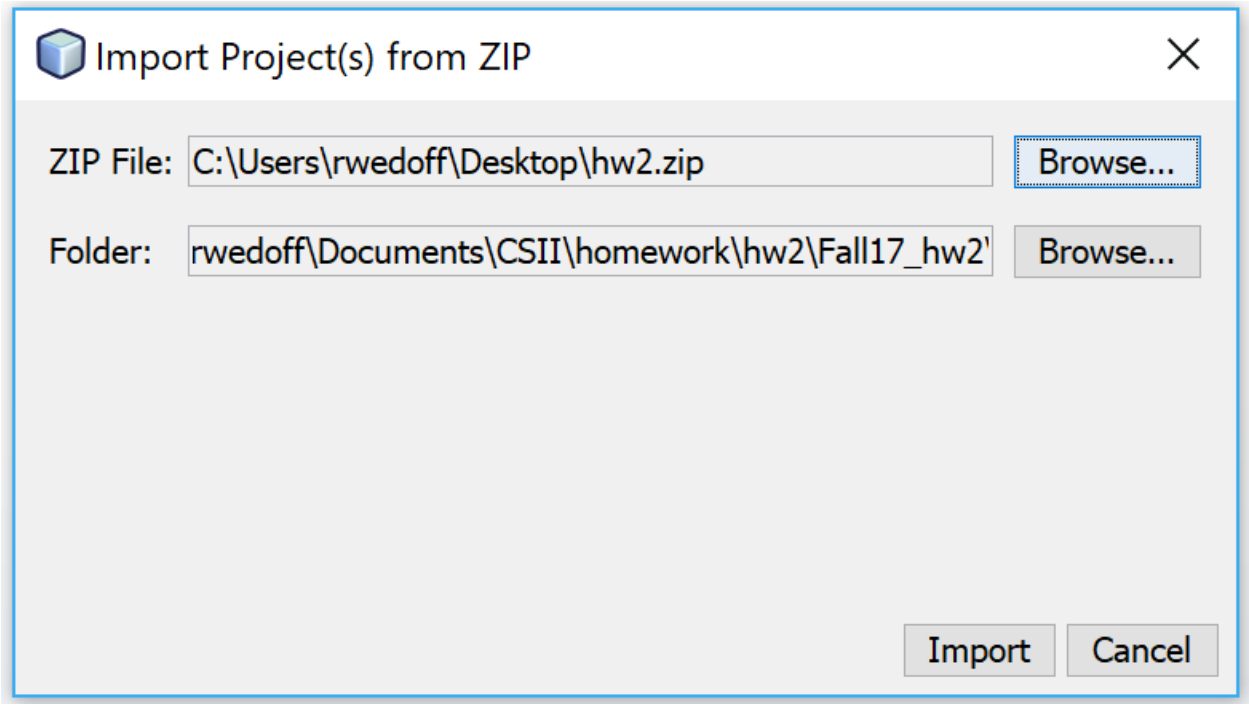
Part 0: Setup the project in NetBeans

If you want to just create a new project and copy/paste the starter code in like HW1, you can do that. Alternatively, the following steps allow you to *import* a project into NetBeans.

1. Open *File -> Import Project -> From Zip*



2.
3. Find the Zip file in the assignment Zip file called hw2.zip.
4. Choose your *Folder* (location where you save your project wisely [recommended, .../Documents/NetBeansProject]
5. Import the project

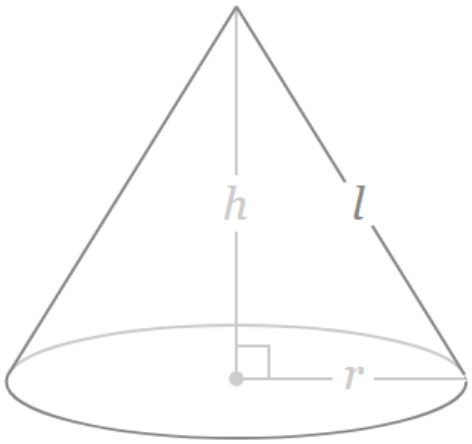


6.
7. Right click on IceCreamShop.java or Set the Run Configuration to run the program. There should be errors at the start of the homework.
8. **Do not change** the package, starter file, or file names. These are required for grading properly.

Part 1: The Cone class

We have provided starter code for a Cone class in Cone.java. A Cone object represents a 3D cone shape, like a ice cream cone (has an open top) with a radius and height.

$$V = \pi r^2 \frac{h}{3}$$



A Cone may be filled with ice cream up to its height. It could be empty, partially full, or completely full. As you'll see, we can scoop ice cream from one Cone into another. After a Cone is constructed, its radius and height never change again, but the height of the ice cream may be increased or decreased by scooping.

Your job is to provide the missing method bodies and methods. We suggest that you work in the following order:

1. Fill in the two constructors. One of them takes two arguments and the other takes zero arguments. You should follow the instructions provided in the comments for each one. In the comments, notice that Cones must always start out with no ice cream in them.

2. Write the methods `getRadius` and `getHeight`. These methods do nothing more than return the value of the corresponding field.

3. Write the methods `getVolume` and `getIceCreamVolume`. The methods return the volume of the Cone and the volume of the ice cream in the Cone, respectively. Recall that the volume of a Cone is $\pi r^2 h/3$. However, to avoid the ugliness of irrational numbers we are going to assume the π is implicit and avoid multiplying it into our answer. Therefore, your methods should just return $r^2 h/3$.

4. We need a way to scoop ice cream from one Cone into another. Fill in the `scoopIceCreamFrom` method. This method takes another Cone called `other` as an argument and moves ice cream from `other` into `this` Cone.

Part 2: The IceCreamShop class

Now that we have working Cones that can hold and scoop ice cream into, let's use several Cones to start an ice cream shop.
This ice cream shop has a lot of indecisive customers. And everyone has a unique order. When a group of customers walk in, they order lots of ice cream, and every customer wants a cone that is a different size (sugar cone, waffle cone, kids cone, cake cone etc.).

You will first add Cones of different sizes to the IceCreamShop.

Second, we give the ice cream shop a full ice cream tub (a large Cone with all the ice cream at first), which will be scooped into the customer's Cones in the order in which the Cones were added to the shop.

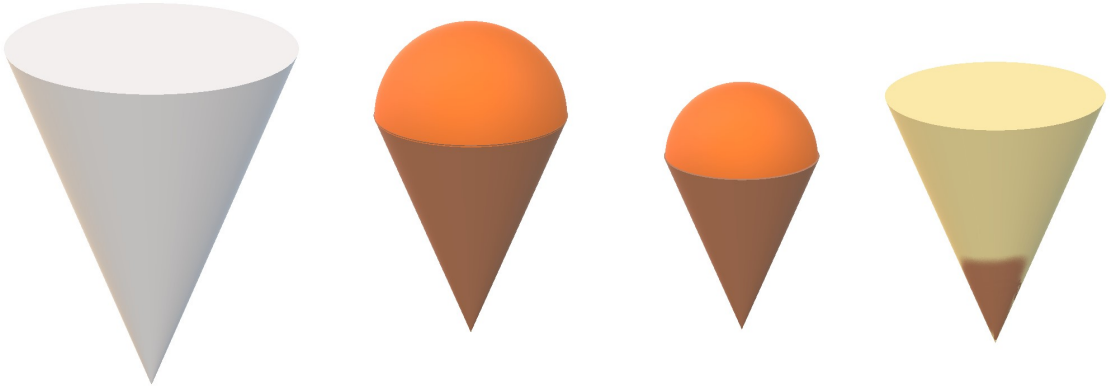
Notice that there can be many kinds of outcomes depending on the sizes of the ice cream tub and the customers' Cones.

Here is an some example scenario. (The ice cream on top is for illustration purposes and does not contribute to the volume). Dark brown is ice cream and light brown is emptiness.

Customer Group Example #1

BEFORE

AFTER



Tub
Vol: 8
Ice Cream Vol: 0

Cone 1
Vol: 4
Ice Cream Vol: 4

Cone 2
Vol: 3
Ice Cream Vol: 3

Cone 3
Vol: 16
Ice Cream Vol: 1

To complete the `IceCreamShop` class, you should do the following.

5. We need a way to add Cones to the shop. Fill in the `addCone` method. Notice that we've already provided two fields for you to use to store the Cones: a Cone array called `cones` and an integer called `count`. If you need an example of how to add a new object to an array, see textbook chapter 3.1 or the lecture notes.

6. We need a way to delete Cones from the game. Fill in the `deleteCone` method. It should delete the Cone at the end of the `cones` array.

7. Finally, you will fill in the `fillConesInOrder` method that scoops ice cream from the ice cream tub to the customers' cones. Use the example customer groups above to make sure you understand the rules. The method must fill the game Cones in the order they were added to the shop.

8. When you've finished the above parts, you should be able to run `IceCreamShop` to see if it works. There are three test cases provided (see the `main` method). Each test case gets a shop, an ice cream tub, and the *expected* volumes at the end of the game.

Helpful Tips

- What do I do if my program prints something like the following?

```
First cone...
Cone(volume=4.0π, iceCreamVolume=4.0π)
Cone(volume=3.0π, iceCreamVolume=3.0π)
Cone(volume=16.0π, iceCreamVolume=1.0π)
Exception in thread "main" java.lang.RuntimeException: failed test: got 1.0 but expected 0.0
        at hw2solution.IceCreamShop.check(IceCreamShop.java:77)
        at hw2solution.IceCreamShop.testCase(IceCreamShop.java:99)
        at hw2solution.IceCreamShop.main(IceCreamShop.java:119)
C:\Users\rwedoff\AppData\Local\NetBeans\Cache\dev\executor-snippets\run.xml:135: The following error occurred while executing this line:
C:\Users\rwedoff\AppData\Local\NetBeans\Cache\dev\executor-snippets\run.xml:118: Java returned: 1
BUILD FAILED (total time: 0 seconds)
```

This message means that the last Cone printed (i.e., `Cone(volume=16.0π, iceCreamVolume=1.0π)` has the wrong ending state. It was supposed to have a ice cream volume of 0π but it only got 1π. You need to debug your program.