CS170 **Lab 8**
Fall 2012 **Methods and Scoping**

Due: Midnight tonight

Objectives of this lab:

- Interpret function calls to determine number and datatype of input parameters as well as datatype of return value
- Write methods utilizing return values and input parameters
- Understand different types of variables and their scopes, including class variables, local variables and parameter variables.
- Practice translating English specifications for programs into algorithms and code.

Lab preparation:

- Start a terminal application and prepare your lab6 directory:

```
mkdir ~/cs170/lab8
cp ~cs170003/share/lab8/*  ~/cs170/lab8
cd ~/cs170/lab8
ls
```

- You should see two files named `Scoping.java` and `TipCalc.java`
- If you do not see these files, ask you TA for help.

Exercise 1: Scoping of variables:

- Open the file `Scoping.java` using gedit:

```
gedit Scoping.java &
```

- Inspect the code in the file. Note that there are places marked in the code with comments like:

```
/* ..................... location (1) */
```

  When "location (1)" is referenced in the instructions or file, it refers to the code to the right of the comment.
- The program contains many different types of variables. In particular:
  - variable a at location (1) is a class variable
  - variable a at location (2) is a parameter variable
  - variable a at location (3) is a parameter variable
  - variable x at location (4) is a local variable
  - variable a at location (5) is a local variable
  - variable a at location (6) is a local variable
- Complete the `F(double a)` method by following the directions given in the TODO comments. Compile and run your code. If it is correct, you should see this output:

```
------ inside F( double a )
a at (2) = -4.44444444E8
a at (1) = 123.0
```

- Write the method calls inside the `{}` after location 5 in the `main` method. If it is not possible to call the requested method, then put the statement:

```
System.out.println("Impossible: Write a method call F()
that ... ");
```
(In other words, you may modify the TODO comment into a printed statement.)

- Next, write the two function calls specified by the TODO statements after location (6) in the `main` method. If it is not possible to call the requested method in the manner specified, then put the statement:
  ```
  System.out.println("Impossible: Write a method call F()
  that ... ");
  ```
  (In other words, you may modify the TODO comment into a printed statement.)
- Complete the function `F(String a)` with the necessary statements.
- Next, write the three function calls specified by the TODO statements after location (7) in the `main` method. If it is not possible to call the requested method in the manner specified, then put the statement:
  ```
  System.out.println("Impossible: Write a method call F()
  that ... ");
  ```
  (In other words, you may modify the TODO comment into a printed statement.)

Exercise 2: Tip Calculation

- Open the file `TipCalc.java` using gedit:
  ```
  gedit TipCalc.java &
  ```
- Read the "TipCalc Problem Description" below. This explains (in English) how this program should work.
- Follow these steps to translate the problem description into algorithms and Java code. Make sure to compile and test after each one.
  - Inspect the `TipCalc.java` program. The main method contains calls to the three functions you will write. You should be able to determine:
    - How many input parameters each function needs
    - What datatype those input parameters should be
    - What return type the function's result should be
  - Using this information, write the three method's headers/signatures.
  - Add a a `return` statement with an appropriate piece of data to the functions that need a return value. This is not to make your methods correct; it is simply to assure that your methods will compile.
  - After you have written your function signatures and return statements, your program should compile. Make sure your program compiles before moving on!!
  - Work on the `bill` function.
    - Remember: this function should calculate the total bill BEFORE tip.
    - What do the input parameters represent?
    - What sort of calculation should this function perform (ie, what is its purpose?)
    - Compile and test to make sure the `bill` function is working correctly.
  - Work on the `tip` function.
    - Remember: this function should compute a 15% tip on the bill.
    - What does the input parameter represent?
    - What sort of calculation should this function perform (ie, what is its purpose?)
    - Compile and test to make sure the `tip` function is working correctly.
  - Work on the `info` function.

- This funtion should print out the information the user needs to know about their bill.
- Sample output with the user entering “4” and “2” for the number of waffles and hashbrowns, respectively:

```
How many waffles? 4
How many hashbrowns? 2
You bought 4 waffles and 2 hashbrowns.
Calculated bill before tip: $16.3
Calculated 15% tip: $2.445
Total bill is $18.745, rounded up to $19.0
```

- All the printing should occur from the `info` function. **Not** the other functions.
- You will need to use a built-in Java `Math` class function to round up to the next dollar. Specifically, the `Math.ceil(double)` function always rounds the input double up to the next integer. For example:

`Math.ceil(5.67)` returns 6.0
`Math.ceil(9.1)` returns 10.0
`Math.ceil(3.4)` returns 4.0

TipCalc Problem Description:

You’ve been up all night playing WOW with your friends. You were so involved in playing WOW that you forgot to have dinner. Since it's nearly 3 am, the only place open is the great institution of Waffle House. You and your friends order some chocolate chip waffles and hashbrowns. The price of 1 chocolate chip waffle is $3.35. The price of 1 plate of hashbrowns is $1.45. However, you have spent your brain's ability to compute simple math on a night of constant video-game awesomeness, and you realize that you just won’t be able to compute the appropriate bill and tip amount for the waitress in your head. For some reason, though, the part of your brain that writes Java functions is super focused and ready to go, so you decide to whip out your laptop and write a Java program to handle the 15% tip calculations for you. (Yes, you brought your laptop to Waffle House with you. You are a computer science student, after all.)

You decide to write 3 functions:

- `bill` which will calculate the total bill for some number of waffles and hashbrowns. The bill does not include the tip.
- `tip` which will calculate the appropriate 15% tip to leave on the total bill.
- `info` which provides the information about the total bill, the tip, and the total amount of money you need to leave. Since you're generous, you always round the total amount up to the next dollar.

Turn-in your work:

- Submit your work using the following commands:

```
cd ~/cs170/lab8
/home/cs170xxx/turnin-lab    Scoping.java       lab8a
/home/cs170xxx/turnin-lab    TipCalc.java       lab8b
```

- You will need to replace the 'xxx' in the above commands with the appropriate section number.
- Make sure you see the “success” message. If you do not, ask your TA for help.