SimpleCalc - Extra Credit

Objective: To implement variables and assignment statements into **SimpleCalc**.

Background:
By now you have a working version of **SimpleCalc** that does basic arithmetic operations. It would be even more powerful if it could store numbers into variable identifiers, then use those identifiers in expressions. For example (user input in **bold**):

```
 -> a = 3
  a = 3.0
 -> b = 4
  b = 4.0
 -> c = a ^ 2 + b ^ 2
  c = 25.0
 -> c
25.0
 -> myId = (28 ^ 2 - 4 / (5 + 3) * 6.5) + 3.4
  myId = 784.15
 -> myId
784.15
```

Identifiers would allow you to store the result of an expression and reuse it in future expressions. You can also provide special constants, like **pi** and **e**.

```
 -> pi
3.141592653589793
 -> e
2.718281828459045
 -> r = 0.5
  r = 0.5
 -> area = pi * r ^ 2
  area = 0.7853981633974483
```

Important!!! Variable names can only contain letters. Digits and other characters are not allowed.

Assignment:
Create a zip file of your original **SimpleCalc** before starting this project.

```
zip -r SimpleCalc.zip SimpleCalc
```

The zip copy will insure you have a working copy saved. (In fact, always archive your files before attempting to modify the code.) Remember, this extra credit project should only be attempted after you have completed the original assignment and thoroughly tested your code.

Adding identifiers requires several steps:

1. Create an **Identifier** class that contains a name (**String**) and a value (**double**). Add the appropriate constructor, getter, and setter methods.

2. Create a database in **SimpleCalc** that keeps track of the **Identifier**s. The easiest implementation is to use an **ArrayList**. To prevent ambiguity, **SimpleCalc** must never allow two **Identifier**s in the database to have the same name.
3. Initialize the variables "**e**" and "**pi**" in your database and set their values to **Math.E** and **Math.PI**
4. Change your user input process to handle the assignment (=) operator. An assignment statement is one in which the first token is an identifier and the second token is the operator "=". The remainder of the tokens are evaluated using your **evaluateExpression** method.
5. Any variable name that has not been assigned evaluates to a 0 value.
6. Add a new user list "**l**" command to display the current variables. For example:

```
 -> l

Variables:
    e               =       2.72
    pi              =       3.14
    r               =       0.50
    area            =       0.79
    a               =       3.00
    b               =       4.00
    c               =      25.00
```

A sample run:

```
% java SimpleCalc
Welcome to SimpleCalc!!!

 -> 5 + 2 * 3
11.0
 -> abba = 5 + 2 * 3
  abba = 11.0
 -> abba
11.0
 -> eightSquared = 8 ^ 2
  eightSquared = 64.0
 -> piTimes2 = pi * 2
0.0
 -> piTimesTwo = pi * 2
  piTimesTwo = 6.283185307179586
 -> l

Variables:
    e               =       2.72
    pi              =       3.14
    abba            =      11.00
    eightSquared    =      64.00
    piTimesTwo      =       6.28

 -> abba
11.0
 -> eddy
0.0
 -> q

Thanks for using SimpleCalc! Goodbye.
```