CT326 Programming III

LECTURE 2

COMMAND LINE PROGRAMMING
COMMON PROBLEMS

DR ADRIAN CLEAR
SCHOOL OF COMPUTER SCIENCE

Objectives for today

- Work through an example of using a text editor and the command line to develop a simple Java program
- Understand some common programming errors and how to address them

Using the Command Line

- We will use the Atom Editor to quickly create and edit sample code during the lectures
  - You should use a full IDE like Eclipse for lab work, etc.
- We will then compile and run the code using command line tools i.e. javac and java
- Need to make sure your runtime environment is setup correctly, especially the CLASSPATH environment variable.
  - OS dependent on how to do this.

In-lecture demonstration

- Account class
  - Has an account number (an int) and a balance (a double)
  - Cannot be created without an account number
  - Customer can deposit an amount to the account and this is added to the existing balance
  - Customer can withdraw a specified amount from the account. It is not possible to withdraw more than the balance
  - It should be possible to query an Account for its balance and account number

Solving Common Coding Problems

- Problem: The compiler complains that it can't find a class.
  - Make sure you've imported the class or its package.
  - Unset the CLASSPATH environment variable, if it's set.
  - Make sure you're spelling the class name exactly the same way as it is declared. Case matters!
  - Also, some programmers use different names for the class name from the .java filename. Make sure you're using the class name and not the filename. In fact, make the names the same and you won't run into this problem for this reason.

Solving Common Coding Problems

- Problem: The interpreter says it can't find one of my classes.
  - Make sure you specified the class name--not the class file name--to the interpreter.
  - Unset the CLASSPATH environment variable, if it's set.
  - If your classes are in packages, make sure that they appear in the correct subdirectory.
  - Make sure you're invoking the interpreter from the directory in which the .class file is located.

Solving Common Coding Problems

- Problem: My program doesn't work! What's wrong with it?
  - Did you forget to use break after each case statement in a switch statement?
  - Did you use the assignment operator = when you really wanted to use the comparison operator ==?
  - Are the termination conditions on your loops correct? Make sure you're not terminating loops one iteration too early or too late. That is, make sure you are using < or <= and > or >= as appropriate for your situation.
  - Remember that array indices begin at 0, so iterating over an array looks like this:

```
for (int i = 0; i < array.length; i++)
    . . .
```

Solving Common Coding Problems

- Are you comparing floating-point numbers using ==?
  - Remember that floats are approximations of the real thing.
  - The greater than and less than (> and <) operators are more appropriate when conditional logic is performed on floating-point numbers.
- Are you having trouble with encapsulation, inheritance, or other object-oriented programming and design concepts?

Solving Common Coding Problems

- Make sure that blocks of statements are enclosed in curly brackets { }.
  - The following code looks right because of indentation, but it doesn't do what the indents imply because the brackets are missing:

```
for (int i = 0; i < arrayOfInts.length; i++)
    arrayOfInts[i] = i;
    System.out.println("[i] = " + arrayOfInts[i]);
```

Solving Common Coding Problems

- Are you using the correct conditional operator? Make sure you understand && and || and are using them appropriately.
- Do you use the negation operator ! a lot? Try to express conditions without it. Doing so is less confusing and error-prone.
- Are you using a do-while? If so, do you know that a do-while executes at least once, but a similar while loop may not be executed at all?

Solving Common Coding Problems

- Are you trying to change the value of an argument from a method? Simple value arguments (like int) are passed by value and can't be changed in a method.
- Did you inadvertently add an extra semicolon (;), thereby terminating a statement prematurely? Notice the extra semicolon at the end of this for statement:

```
for (int i = 0; i < arrayOfInts.length; i++) ;
    arrayOfInts[i] = i;
```

Next time…

- Testing and Test Driven Development