CS102
Introduction to data structures, algorithms, and object-oriented programming

DAY 2
DrJava

- This IDE is available on all the networked CS machines.
- It is also freely downloadable from a site you will find if you google DrJava.
- If you have trouble installing DrJava, see me or one of the coaches. The install contains the Java JDK and a terminal application.
Lab 1 recap

• “pico &” puts your session into the background and you need to type “fg” to get it back on the screen.

• I changed the name of the file you were editing in the directions from MyWorld.java to MyHello.java (oops!)

• In the lab, I intended to give you an idea of how java programs can be compiled and run both inside DrJava and at the command line.
Java Code

- Every Java application begins with a class header statement such as:

```java
public class ClassName {
    // a ‘}’ opens a new block of code, ‘}’ ends it.
    All code written in a Java program, except for global variable declarations, and import statements must be written inside methods, which must be written inside classes
}
```
Chapter 1 Synopsis

Chapter 1 of Eck book starts with discussion of components in a computer and how operating systems work (in general)

- Sect. 1.3 describes how Java programs are compiled and interpreted.

- Sect. 1.4 talks about basic aspects of programming: data (variables and types) and instructions (control structures and subroutines).

- Sect. 1.5 introduces top-down and bottom-up design and class hierarchies.

- Sect. 1.6 gives a peek at graphical user interfaces and 1.7 briefly touches on network protocols.
Structures in a Java program

A Java program (i.e., class) is either:

• a library of static methods (functions) that may return values or just have side effects; or

• a data type definition (sort of like using define-struct)

There is one static method that must be included in every set of Java programs: the main method.

Each program starts execution at a method with the following signature:

public static void main(String[] args){...}
Method signatures

Methods always contain code inside a set of {}s. The first line of a method is called the signature. The signature tells the programmer everything they need to know about how to use the method.

Form of main method signature:

    public static void main(String[] args)

Over the course of the semester, we’ll learn what public and static in the signature mean. For now, concentrate on the parts in color.
Method signatures

Form of main method signature:

    public static void main(String[ ] args)

void: the return type, in this case, nothing.

main: the method name

String[ ] args: the parameter type(s) and name(s)

Most code must be written inside methods, which are delimited by { }
A data type is a set of values and a set of operations on those values.

There are four primitive data types that can be considered the basis of the Java language:

1. Integers, with arithmetic operations (int)
2. Real numbers, with arithmetic operations (double)
3. Booleans, with logical operations (boolean)
4. Characters, alphanumeric symbols (char)

Expressions are sequences of text in which operations are applied to a data type.
Java Naming Conventions

Classes, variables, methods, and constants are each identifiers named according to programmer conventions.

• Class names should start with a capital letter and all multiple-word names should have each word start with a capital letter.

• Variable and method names are lowercase except if they have multiple words, in which case the start of each word should be capitalized.

• Constants should be written in all capital letters, with words separated by _ (underscore).
Java Statements

Statements:
  a. variable declarations,
  b. assignments,
  c. conditionals (if, else), **
  d. loops (for, while, do-while), **
  e. method calls, and
  f. returns.

Each statement must end with a ; (semicolon) except those marked with ** above. Each of the ** statements should have { } surrounding inner statements...the inner statements inside end in ;

 NEVER put a ; after a }
Java Keywords

Keywords are reserved, meaning that you can’t use them for identifier names. The complete list can be found at:

http://docs.oracle.com/javase/tutorial/java/nutsandbolts/_keywords.html

Java has more keywords than Racket.

All keywords are entirely lowercase. Don’t try to memorize the list.
Java Statements

Statements:

a. variable declarations,
b. assignments,
c. conditionals (if, else),
d. loops (for, while, do-while),
e. method calls, and
f. returns.
Declaration statements

All variables have to be declared as a particular type before they can be used in an expression.

An expression is a combination of a type and the operators that can be used on that type.

Variables declared inside a method block are called “local variables”.

For example, to use the integer variable i in a method, the use must be preceded by the line

```
int i;
```
Java Statements

Statements:

a. variable declarations,

b. assignments,

c. conditionals (if, else),

d. loops (for, while, do-while),

e. method calls, and

f. returns.
Assignment Statements use =

A primitive variable name is actually a memory location. The value you assign to that name becomes the content of that memory location.

For example, after declaring the integer \( i \) you could assign a value (sometimes a literal, sometimes an expression) to \( i \) as follows:

\[
i = 5;
\]

You can also combine declaration and assignment as follows:

\[
\text{int } i = 5;
\]
We already saw string literals used in lab.

Numeric literals are discussed in Eck, Sect 2.2.3

Character literals are written between apostrophes: ‘A’, ‘b’, ‘\n’ etc.
Java Statements

Statements:

a. variable declarations,

b. assignments,

c. conditionals (if, else),

d. loops (for, while, do-while),

e. method calls, and

f. returns.
Method calls

Like calling a function in Racket. A method call always has the form:

```
methodName(argument list);
```

For example, let’s look at some Java code.