Lecture 2
CSE11 Spring 2015

Outline

- Compiling and Running Java Applications
- Primitive Variables/Assignment (ZY:Ch 2)
- Java Identifiers
- Naming conventions
- Working with Numbers
- Precedence

What is a computer “program”?

- A sequence of steps (instructions) to perform a specific task
    - Sum a column of numbers
    - Store and retrieve names/addresses from a phonebook
    - Run the fuel-injection sequence in your car
    - Simulate weather for tomorrow.
    - Send text messages
    - ….
- Computers themselves only understand a very small number instructions
    - E.g., ARM Processors (your Android Cell Phone) have 6 different types of instructions. Fewer than 100 total instructions.

Hello World!

- Traditional First Application (HelloWorld.java)

```
public class HelloWorld {

public static void main(String[] args) {
          System.out.println("Hello, World");
    }

}
```

- Compile and run

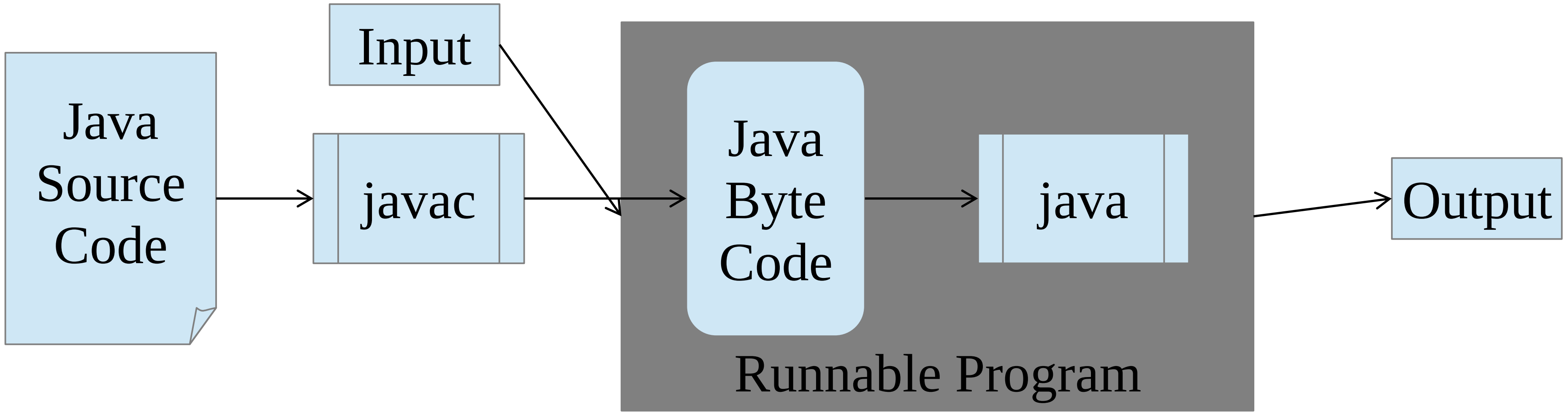
```
$ javac HelloWorld.java
$ java HelloWorld
Hello, World
$
```

Java is a compiled language

- Source code must be converted into a form that the CPU understands (machine instructions or machine code)
- Interpreters
  - Perform this conversion when the code is run.
    - Python, Perl. BASIC are examples
- Compilers
  - Perform this conversion before the program is run
  - Output is called a “binary”
  - Java, C, C++, FORTRAN, Pascal are examples

Java Compiler

- javac – translates source code to java byte code (this is a form of a binary)
- java – Java Virtual Machine (JVM), interprets java byte code
- JVMs are specific to each physical CPU/OS
- Byte code runs on any (compliant) JVM



Remember!

- The CPU only understands binary
- It’s the job of the compiler to get source code into a binary format that the computer can understand
- Java compiles to a universal binary (java byte code)
- The Java Virtual Machine (JVM) is (another) binary that runs java byte code

Some Java Rules

1. A Java source file may have only one class declared as <u>public</u>
2. Names of variables, classes are case sensitive. helloworld != HelloWorld
3. If a source file has a public class defined, the file must be the name of the public class.
   1. E.g. HelloWorld.java
4. The method called main, is special. It must be declared: `public static void`

(We will clarify all of these keywords: `public`, `class`, `static`)

Variables/Identifiers

- A unit of storage.
- Suppose you are summing a column of numbers
  - Need a variable (storage) to keep track of computation (partial sum)
    - Call it something in your program, e.g. `columnSum`
    - It will consume memory
    - It has a **type**
- Type
  - An integer number is different than a decimal (floating point) number
  - At the machine level
    - Two different ADD instructions. One that adds integers, another one that adds floating point numbers

Primitive Types

type	Size (bits)	Min Value	Max Value
`byte`	8	-127	128
`short`	16	-32767 (-(2^{15}-1)	32768 (2^{15})
`int`	32	-(2^{31}-1) ~ 2Billion	2^{31} ~ 2Billion
`long`	64	-(2^{63}-1)	2^{63}
`float`	32	~7 digits of accuracy. 1 byte exponent	
`double`	64	~15 digits of accuracy. 2 byte exponent	
`boolean`	unspecified	True or False	
`char`	16	0000 (HEX)	FFFF (HEX)

- These are the data types understood directly by CPUs
- Every other type in Java is called an **Object Type**

- **Java treats object types and primitive types differently** in a variety of circumstances

Overflow

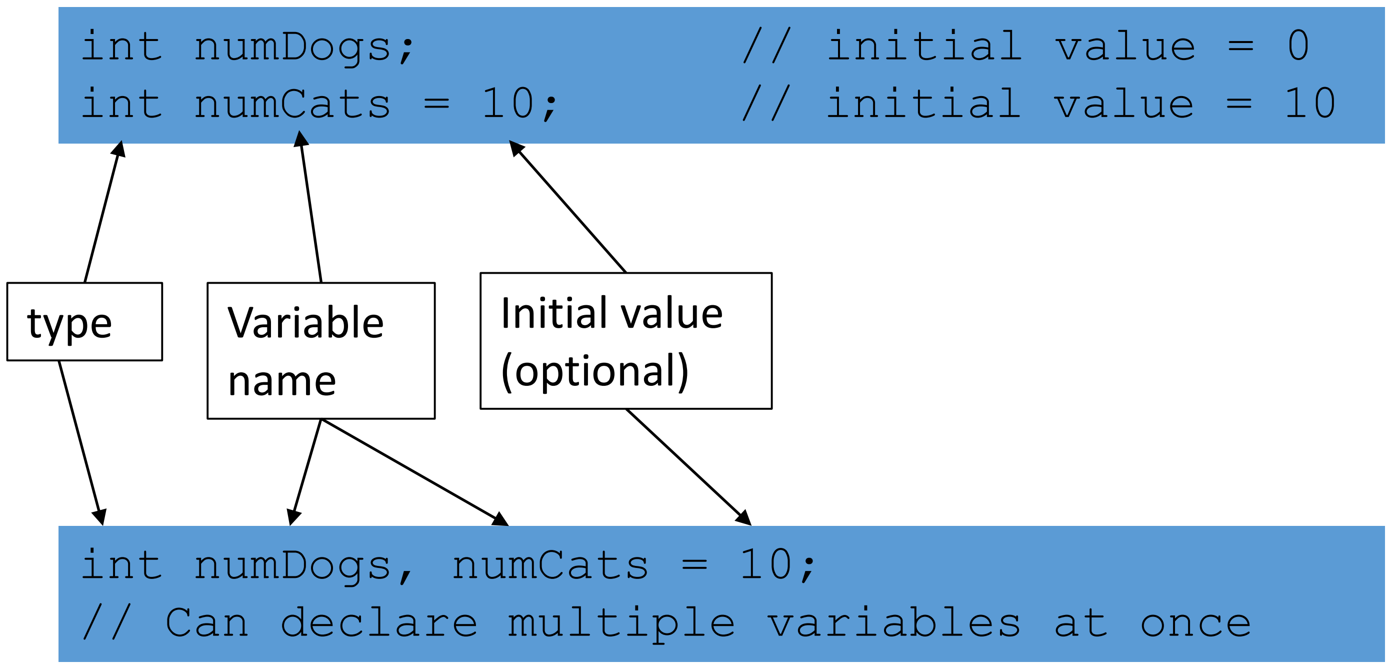
- Different variable types have finite precision and maximum/minimum values
- If one attempts to store a value too large into a variable it is called an overflow
    - `short myshort = 1048576;` is an overflow

Java Names

- Valid Identifiers (variable names, class names, method names, parameter names)
- Series of letters, numbers, underscores,$ (a-z)(A-Z)(0-9)(“_”)(“$”)
- An identifier **CANNOT begin** with a number.
- An Identifier CANNOT have other special characters.
- Limit the use of $, in identifiers. While legal, it is generally reserved for automatically-generated code.

Declaring Variables

- You must declare (define) a variable before you can use it in your program



```
int numDogs;                // initial value = 0
int numCats = 10;           // initial value = 10
```

```
int numDogs, numCats = 10;
// Can declare multiple variables at once
```

Test understanding

Declare a variable `classUnits` initialized to 4

```
int classUnits = 4;
```

Declare a variable `grade` initialized to ‘A’

```
char grade = 'A';
```

Declare a variable `gpa` initialized to 3.68

```
float gpa = 3.68;
```

Numeric Expressions

- An expression is formula or calculation.
  - It’s result can be stored in a variable
  - Or used as argument to function/method/subroutine
- A numeric expression is just like what you write down grade school
  - can use declared variables in the expression
  - `( 10 + 5 )/3`
  - `3*numCats + numDogs`
- Assignment
  - `int numPets = numCats + numDogs;`

Type Conversions

- Variables of one type **cannot be assigned** to a variable of a different type
- Variables can be **converted** from one type to another before assignment.
- The programmer can explicitly declare a conversion with a cast (suppose `numDogs` is an `int`)
  - `float fdogs = (float) numDogs;`
- Java will do implicit conversions for you, be careful!

How Java deals with number types in expressions?

If ALL of the components are integers, the expression is integer

If ANY of the components are double, then the expression is double

- 1 + 2 + 3 + 4 (integer)
- 1 + 2.0 + 3 + 4 (double)

Precedence of Arithmetic Operations

Follows rules of grade-school arithmetic

- multiplication, division first

- addition, subtraction second

- when equal precedence operators bind left to right

What is:

3 * 6 / 2 * 8 = (3 * 6)/(2*8) or ((3*6)/2) * 8?

Use parentheses when a statement looks ambiguous.

Math Operators

operand1 $\boxed{}$ operand2

Operator	result with integer operands	result with at least one float operand
`+`	`int`	`float`
`-`	`int`	`float`
`*`	`int`	`float`
`/`	`integer division`	`float`
`%`	`remainder operator*` `(AKA mod or modulus operator)`	`float`

% -- in some other languages (e.g. C) only defined for integer operations.

More advanced Computations: Math Methods

- If we could only perform basic math, programs would be too restricted.
  - What if you want square roots, cos, tangent, natural log?
- Methods (aka subroutines or subprograms)
  - Take 0 or more input arguments
  - Return or calculate an output
  - Arguments and output are typed.
- java.lang.Math
  - A “package” of methods that perform different calculations
  - e.g., $\sqrt{x}$ can be computed with `Math.sqrt(x)`
    - `x` can be an `integer` or `double`
    - Always “returns” a `double`
    - See http://docs.oracle.com/javase/7/docs/api/java/lang/Math.html

Subroutine/Method Return Values

- All methods return "something". That something is of a particular **type**
  - `void`: this is the empty type. Means that nothing is returned by the method
  - primitive types: when a primitive type is returned, a value of that type is returned
  - object references: can return a reference to any java Object. Will cover in detail over the next few lectures
- Using a method within an expressions

```
float dia, area = 36.0;
dia = 2 * Math.sqrt(area/Math.PI);
```

Putting things together

```
import java.lang.Math;
int myInt = 109;
double result;
result = Math.sqrt(myInt);
```

$\uparrow$
argument
(or parameter)

Definition of `Math.sqrt` says that it's argument must be a `double`. myInt is of type `int`

This is an <u>implicit type conversion</u>. The java compiler knows that it can convert an integer to type double without loss of precision. It does so, silently.

When complete, the variable result will hold the value of $\sqrt{109} \approx 10.4403$

Random Numbers

- There are times when a random number (or random bits) are required.
- Cryptography, games, testing, etc.
- Mathematically, It is quite difficult for a computer to generate a truly random sequence of bits (numbers)
- Libraries (pre-defined methods) can be used to generate pretty good random sequences
  - Pretty good means that any combination of bits is about as likely as any other combination of bits

What is pseudorandom

- Random number generators are *seeded* with a starting number
  - If you give the same seed, you will get the identical sequence of “random” numbers every time you run the program.
    - Called a pseudorandom sequence
- What happens if you run the following code repeatedly?

```
import java.util.Random;
 …
Random myRnd= new Random();
Random seedRnd= newRandom(10);
System.out.println(myRnd.nextInt());
System.out.printlin(seedRnd.nextInt());
```

Defining Numerical Constants

- Very often, one wants a numerical constant defined a program
  - Math package defines π (Math.PI) and e (Math.E) as high-precision constants.
  - Programmers can (and should) define constants when appropriate
- `final` is the keyword to define a variable as not changeable
- By convention, constant variables are written in `ALL CAPS`

Naming Conventions

- **Class/interface** – Begin with Upper case, then mixed. **Pronouns or Nouns**
  - e.g. RisingSun, Bicycle, Animal
- **Method,** start with lower case, then mixed (AKA camel case). Usually a **verb/action**
  - setColor(), getHeight(), multiply()
- **Instance and class variables**. Start with lower case, then mixed. **Short nouns, phrases**
  - lastPosition, radius, backgroundColor
- **Local variables/parameters**. Lower case letters, **short words, abbreviations**
  - cnt/count, prev/previous, nxt/next, len/length
- Constants – ALL CAPS with Underscores
  - MAX_HEIGHT, MIN_WIDTH, NUM_SEATS

Comments

Comments are *essential*. They describe in English what a class is supposed to represent or how a particular algorithm works

// Anything after until new line is a comment

/* Anything between the opening slash-asterisk and closing slash-asterisk is a comment. This is how multi-line comments are entered */

Strings

- Strings: “this is a string”
  - A sequence of characters
  - NOT a primitive type
- We will do many things with Strings throughout the course.
- Let’s use the String type to practice reading java documentation.
- http://docs.oracle.com/javase/7/docs/api/java/lang/String.html