Java Data Types

• Primitive types. boolean, byte, char, short, int, long, float, double

• Reference Type. Class.
boolean

• 1 bit of data. Can only be of two value, *true* or *false*
• Declaration: boolean b;
• Assignment: b = true; b = false;
• *true* or *false* are Java constants. They are not the same as string “true”, or “false”, or 1, or 0, or zero, or nonzero.
• “cast”: force conversion from one type to another type. Java is a strong typed language. “cast” means “True me! I know what I am doing!”
Demo DataTypes.java

- We will revisit this code many times.
byte & char

• byte: 1 signed byte. Each byte is 8 bits. – 128 to 127.
• Char: 2 unsigned bytes. Unicode, 0 to 65535
short, int, long

- short: 2 signed bytes, -32768 to 32767
- int: 4 signed bytes, -2147483648 to 2147483647
- long: 8 signed bytes, -9223372036854775808 to +9223372036854775807
float & double

- float: 4 signed bytes, Covers a range from 1.40129846432481707e-45 to 3.40282346638528860e+38 (positive or negative).
- double: 8 signed bytes, Covers a range from 4.94065645841246544e-324d to 1.79769313486231570e+308d (positive or negative).
Cast

- *Cast* is a forced conversion from one data type to another data type.
- Java is a strong typed programming language. Certain data type conversions are now allowed, others are done through *cast*.
- Some numeric data types can be cast into other numeric data types. Numeric data types are Byte, short, int, long, float, and double. However data loss may occur. Some numeric data type conversions may result in errors.
Cast

• Certain cast is not allowed. E.g. boolean cannot be cast to numeric types.
• Homework exercise: test different cast situation, what is allowed, what is not.
Arithmetic Operators

• Addition: +
• Subtraction: -
• Multiplication: *
• Division: /
• Remainder or modulus: %
Demo arithmetic.java
Learn through practice

• Now modify the data type of the arithmetic.java program and see what happens. Can you explain it.

• Modify the sample, even break it. Use Eclipse to learn the reason why you break the code. Find resources from javasoft website, do a google search, find the answer. Then learn how to correct your code.
Rules of operator precedence

- Multiplication, division and remainder operations are applied first.
- Addition and subtraction operations are applied next.
- If an expression contains several such operations, the operators are applied from left to right.
- When there is parentheses, the expression with it is evaluated first.
Example

- int a = 1;
- int b = 2;
- int c = 3;
- int d = 4;
- int e = a + b * c - d = ?
- int f = (a + b) * (c - d) = ?
Reading Assignments

- “Head First Java” p47 – 51
- “Java2” p35- 54, p58 - 68