

ITEC1620
Object-Based Programming

Lecture 5
Working with Data

Declaring Variables

<datatype><identifier / variable name>;
int myNumber;

- Declaring a variable does three things
 - Defines a new identifier
 - Specifies its datatype
 - Allocates / binds it to a space in memory

Primitive datatypes

- byte - 8 bits
- short - 16 bits
- int - 32 bits
- long - 64 bits
- float - 32 bits
- double - 64 bits
- boolean
 - true / false
- char
 - 16 bits (unicode)

Declaration and Assignment

- Declaration allocates spaces in memory

myNumber



- Assignment copies values into them

myNumber = 5;

myNumber



Declaration and Assignment II

int anotherInt = 7;

anotherInt

7

double firstFloat = 1.6;

firstFloat

1.6

double secondFloat = firstFloat;

secondFloat

1.6

Style

- All identifiers in JAVA have a proper style guide/naming convention
- Identifiers that are not loop counters or math symbols should never be a single character
- Identifiers should be all lower case except for the first letter of each additional word
- Identifiers should not use underscores
- Identifiers cannot start with numbers and cannot use most non-alphanumeric symbols

Computer Assignment vs. Mathematical Equating

- Assignment copies values, it does not equate symbols

firstFloat = 0.2;

firstFloat

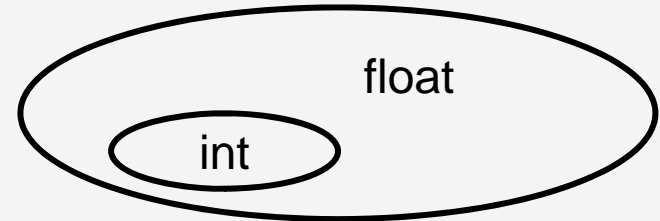
0.2

secondFloat

1.6

Data Conversions

- int is a subset of float
 - Any int is a float
 - Java perform automatic conversions for “widening” assignments
 - int to float, float to double, etc



`firstFloat = myNumber;`

`firstFloat`

5.0

Data Conversions II

- float is a super-set of int
 - A float may or may not be an int
 - If not, data is lost
 - Java requires an explicit request to do conversions for “narrowing” assignments
 - float to int, double to float, etc
- `myNumber = (int) secondFloat;`

myNumber

1

Notes on Data Conversions

- Conversions to super-sets
 - Automatic in Java (cast is optional)
- Conversions to sub-sets
 - Syntax error if no cast
 - Conversion by “truncation” – chop excess
- No conversions between disjoint sets
 - e.g. int and boolean



Questions?

Mathematical Operators

- Unary operators
 - + (positive), - (negative)
- Binary operators
 - *, /, % (modulus), + (add), - (subtract)
- Assignment
 - =
- Normal precedence rules

Integer Division

- If both operands are integers, the result of / is also an integer
 - $2 / 3 \rightarrow 0$
 - $4 / 3 \rightarrow 1$
 - $257 / 100 \rightarrow 2$
 - $257 / 10 \rightarrow 25$
- Internally, create a double and truncate

Floating Point Division

- At least one of the two operands must be a floating point value (i.e. float or double)
 - $2 / 3.0$
 - $2.0 / 3$
 - $2.0 / 3.0$

Floating Point Division II

- Note: floating point division produces a double

```
float aFloat = 4.0 + 2 / 3.0;    //error
```

```
float aFloat = (float) (4.0 + 2 / 3.0);
```

```
double aDouble = 4.0 + 2 / 3.0;
```

Floating Point Division III

double first = 1.0;

double second = 1/3.0 + 1/3.0 + 1/3.0;

- Are first and second equal?
- What is $0.33 + 0.33 + 0.33$?

double third = 1/2.0 + 1/2.0;

- Are first and third equal?

Modulus - %

- Remainder after integer division
 - $5 \% 3 = 2$
 - $4 \% 2 = 0$
- Divide students into 10 groups based on student number
 - `studentNumber % 10` → value from 0-9

Using Modulus

- Finding multiples
 - if $x \% y == 0$, x is a multiple of y
- Even-odd numbers
 - if $x \% 2 == 0$, x is even
 - if $x \% 2 == 1$, x is odd
- Making bins
 - n bins numbered from 0 to $n-1$
 - mod by n

Precedence

- `int first = 2 + 4 * 3;`
- `int second = (2 + 4) * 3;`
- `int third = (2 + 4) / 3;`
- `int fourth = 2 + 4 / 3;`
- `float fifth = 2 + 4 / 3;`

Precedence Rules

- Unary operators +, -
- Binary operators *, /, %
- Binary operators +, -
- Assignment =

- float fifth = 2 + 4 / 3;
 - Integer division, then assignment (cast)

Other Operators

- `+=`, `-=`, `*=`, `/=`, `%=`
- `var = var + something;`
 - Happens so often that JAVA provides a short cut
 - `var += something;`
 - Update the value of var by adding, subtracting, etc something with it

Other Operators II

- `var += 1;`
 - Happens so often that JAVA provides a short cut
 - `var++;`
 - or
 - `++var;`
- Difference is whether increment and update occur before or after access
 - Avoid using increment operator except on a line by itself – too easy to make errors

Questions?

Evaluation Sample

```
int x = 0;
for (int i = 0; i < 10; i++)
{
    if (i % 5 EQUALS 2)
        x += 50;
    if (i % 3 EQUALS 2)
        x += 30;
}
```


Evaluation Sample II

```
int x = 0;
int y = 25;
while (y <= 50 AND x <= 50)
{
    if (y > x)
        x += 20;
    else
        y += 25;
}
```

Example Program

- Write a program fragment that calculates the sum of the digits of int value input by the user
 - `// int input = ...;`
 - `input = 1234`
 - Result – 10
 - `input = 562`
 - Result – 13
 - `input = 26`
 - Result – 8

Primitive datatypes are ... Primitive!

- Primitive data
 - int, float, char
- Real world data
 - Bank statement, transcript, billing address
- Need to create “aggregate” structures (i.e. user defined datatypes)

Readings and Assignments

- Text sections (5th, 6th, or 7th edition)
 - 2.2-2.5
- Tutorial – Evaluating JAVA
- Lab Assignment 2