6.092: Intro to Java

2: More types, Methods, Conditionals
Outline

- Lecture 1 Review
- More types
- Methods
- Conditionals
Variables

Named location that stores a value

Form:

\[ \textit{TYPE NAME}; \]

Example:

\[ \text{String foo;} \]
Types

Limits a variable to kinds of values

**String**: plain text ("hello")

**double**: Real numbers (3.14)
Operators

Symbols that perform simple computations

Assignment: =
Addition: +
Division: /
Assignment 1 Review
Assignment 1 Review

```java
class TempConverter {
    public static void main(String[] arguments) {
        double input = 90;
        double celsius = (5/9.0)*(input-32);
        System.out.println("The value is " + celsius + "C");
    }
}
```
Outline

- Lecture 1 Review
- More types
- Methods
- Conditionals
More types

String: plain text ("hello")
double: Real numbers (3.14)
int: integer (5, -18)

Division ("/") operates differently on integers and on doubles!
Order of Operations

Precedence like math, left to right
Right hand side of = evaluated first

double x = 3 / 2 + 1; // x = 2.0
String Operators and Conversions

- Concatenation: +

```java
String text = "hello " + "world";
String text = text + " number " + 5;
// text = "hello world number 5"
```
String Operators and Conversions (c’ed)

- Don’t mess with types!

```java
String five = 5; // not good!
```

```java
test.java.2: incompatible types
found: int
required: java.lang.String
String five = 5;
```
Conversion by casting

```java
int a = 2;    // a = 2
double a = (double)2;  // a = 2.0

double a = 2/3;    // a = 0.0
double a = (double)2/3;  // a = 0.6666...

int a = (int)18.7;    // a = 18
```
Conversion by method

int to String:
String five = Integer.toString (5);
String five = "" + 5; // five = "5"

String to int:
int foo = Integer.parseInt ("18");
Mathematical Functions

Math.sin(x)

Math.cos(Math.PI / 2)

Math.log(Math.log(x + y))
Outline

- Lecture 1 Review
- More types
- Methods
- Conditionals
Adding Methods

```java
public static void NAME() {
    STATEMENTS
}
```

To call a method:

```java
NAME();
```
class NewLine {
    public static void newLine() {
        System.out.println("\n");
    }

    public static void threeLines() {
        newLine(); newLine(); newLine();
    }

    public static void main(String[] arguments) {
        System.out.println("Line 1");
        threeLines();
        System.out.println("Line 2");
    }
}
Parameters

public static void \textit{name}(\textit{type name}) {
  \textit{statements}
}

To call:

\textit{name} (\textit{expression}) ;
class Square {
    public static void printSquare(int x) {
        System.out.println(x*x);
    }

    public static void main(String[] arguments) {
        int value = 2;
        printSquare(value);
        printSquare(3);
        printSquare(value*2);
    }
}
class Square2 {
    public static void printSquare(int x) {
        System.out.println(x*x);
    }

    public static void main(String[] arguments) {
        printSquare("hello"); // not good!
        printSquare(5.5);
    }
}
class Square3 {
    public static void printSquare(double x) {
        System.out.println(x*x);
    }

    public static void main(String[] arguments) {
        printSquare(5);
    }
}
Multiple Parameters

[…]

NAME(TYPE NAME, TYPE NAME) {
  STATEMENTS
}

NAME(arg1, arg2);
class Multiply {
    public static void timesRoot(double a, double b) {
        System.out.println(Math.sqrt(a * b));
    }

    public static void main(String[] arguments) {
        timesRoot(2, 2);
        timesRoot(3, 4);
    }
}
Return Values

public static TYPE NAME() {
    STATEMENTS
    return EXPRESSION;
}

void means “no type”
class Square4 {
    public static int square(int x) {
        return x * x;
    }
}

public static void main(String[] arguments) {
    System.out.println(square(5));
    System.out.println(square(2));
}
}
Variables in Methods

Variables live in the block ({}) where they are defined (scope)

Parameters are like defining a new variable in the method
class SquareChange {
    public static void printSquare(int x) {
        System.out.println("printSquare x = " + x);
        x = x * x;
        System.out.println("printSquare x = " + x);
    }

    public static void main(String[] arguments) {
        int x = 5;
        System.out.println("main x = " + x);
        printSquare(x);
        System.out.println("main x = " + x);
    }
}
Methods: Building Blocks

- Big programs are built out of small methods
- Methods can be individually developed, tested and reused
- User of method does not need to know how it works
- In CS, this is called “abstraction”
Outline

- Lecture 1 Review
- More types
- Methods
- Conditionals
if statement

if (COMPARISON) {
    STATEMENTS
}

class If {
    public static void test(int x) {
        if (x > 5) {
            System.out.println(x + " is > 5");
        }
    }
    public static void main(String[] arguments) {
        test(6);
        test(5);
        test(4);
    }
}
Comparison operators

x > y: x is greater than y
x < y: x is less than y
x >= y: x is greater than or equal to y
x <= y: x is less than or equal to y
x == y: x equals y (assignment: =)
Comparison operators

- Do NOT call `==` on doubles! EVER.

double a = Math.cos (Math.PI / 2);
double b = 0.0;

a = 6.123233995736766E-17
a == b will return FALSE!
else

if (COMPARISON) {
    STATEMENTS
}
else {
    STATEMENTS
}
}
public static void test(int x) {
    if (x > 5) {
        System.out.println(x + " is > 5");
    } else {
        System.out.println(x + " is not > 5");
    }
}

public static void main(String[] arguments) {
    test(6);
    test(5);
    test(4);
}
else if

if (COMPARISON) {
    STATEMENTS
} else if (COMPARISON) {
    STATEMENTS
} else if (COMPARISON) {
    STATEMENTS
} else {
    STATEMENTS
}
public static void test(int x) {
    if (x > 5) {
        System.out.println(x + " is > 5");
    } else if (x == 5) {
        System.out.println(x + " equals 5");
    } else {
        System.out.println(x + " is < 5");
    }
}

public static void main(String[] arguments) {
    test(6);
    test(5);
    test(4);
}
class Scope {
    public static void main(String[] arguments) {
        int x = 5;
        if (x == 5) {
            int x = 6;
            int y = 72;
            System.out.println("x = " + x + " y = " + y);
        }
        System.out.println("x = " + x + " y = " + y);
    }
}
Assignment: FooCorporation

Method to print pay based on base pay and hours worked

Overtime: More than 40 hours, paid 1.5 times base pay

Minimum Wage: $8.00/hour

Maximum Work: 60 hours a week
Reminder

- Write **your own** code

- Homework due tomorrow (Friday) 7pm on Stellar.