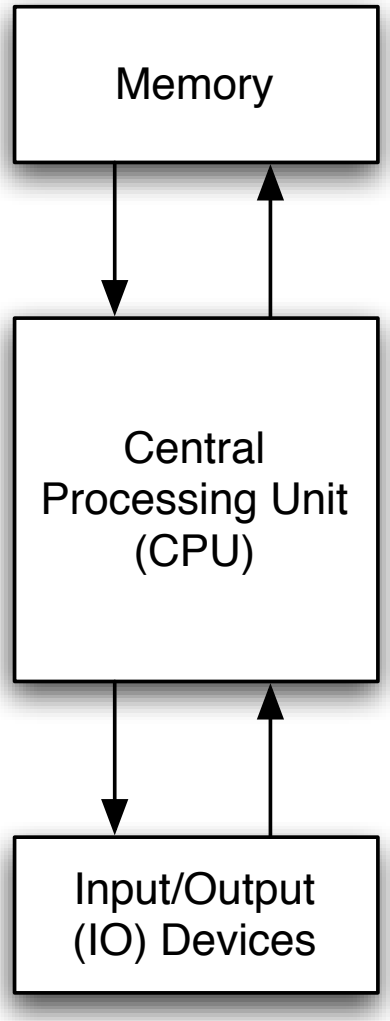
6.092: Introduction to Java

Lecture 1: Types, Variables, Methods

Outline

- Intro to Java
- Types and variables
- Operators
- Methods

The Computer



CPU Instructions

z = x + y

Read location x

Read location y

Add

Write to location z

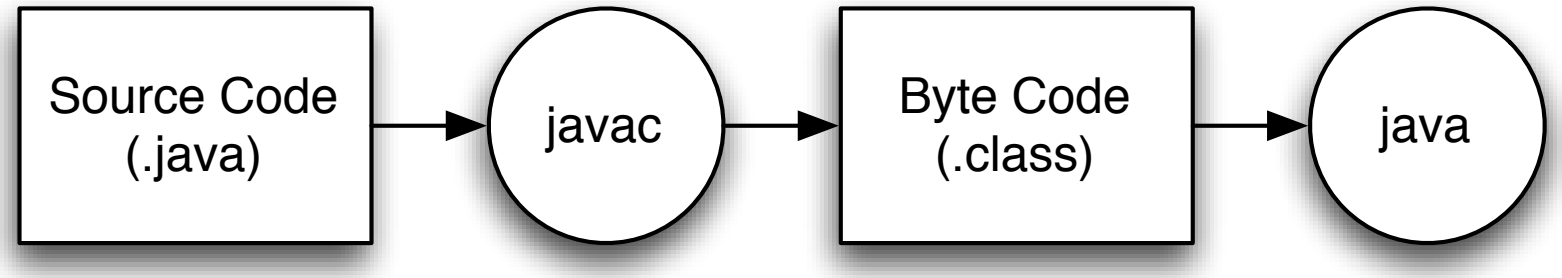
Programming Languages

- Easier to understand than CPU instructions
- Needs to be translated for the CPU to understand it

Java

- “Most popular” language
- Runs on a “virtual machine” (JVM)
- More complex than some (eg. Python)
- Simpler than others (eg. C++)

Compiling Java



Program Structure

```
class CLASSNAME {
    public static void main(String[] arguments) {
        STATEMENTS
    }
}
```

Outline

- Intro to Java
- Types and variables
- Operators
- Methods

Types

Kinds of values that can be stored and manipulated.

boolean: Truth value (**true** or **false**).

int: Integer (0, 1, -47).

double: Real number (3.14, 1.0, -2.1).

String: Text (“hello”, “example”).

Variables

Named location that stores a value of one particular type.

Form:

TYPE NAME;

Example:

String foo;

Assignment

Use = to give variables a value.

Example:

```
String foo;
foo = “IAP 6.092”;
```

Assignment

Can be combined with a variable declaration.

Example:

```
double badPi = 3.14;
boolean isJanuary = true;
```

Output

System.out.println(*some String*) outputs to the console

Example:

```
System.out.println("output");
```

```
class Hello3 {
    public static void main(String[] arguments) {
        String foo = "IAP 6.092";
        System.out.println(foo);
        foo = "Something else";
        System.out.println(foo);
    }
}
```

Outline

- Intro to Java
- Types and variables
- Operators
- Methods

Operators

Symbols that perform simple computations

Assignment: =

Addition: +

Subtraction: -

Multiplication: *

Division: /

```
class DoMath {
    public static void main(String[] arguments) {
        double score = 1.0 + 2.0 * 3.0;
        System.out.println(score);
        score = score / 2.0;
        System.out.println(score);
    }
}
```

Division

Division (“/”) operates differently on integers and on doubles!

Example:

```
double a = 5.0/2.0; // a = 2.5
int b = 4/2; // b = 2
int c = 5/2; // c = 2
double d = 5/2; // d = 2.0
```

Mismatched Types

Java verifies that types always match:

```
String five = 5;  // ERROR!
```

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

Conversion by casting

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

int a = 18.7;        // ERROR
int a = (int)18.7;       // a = 18

double a = 2/3;     // a = 0.0
double a = (double)2/3;  // a = 0.6666…
```

Outline

- Intro to Java
- Types and variables
- Operators
- Methods

Methods

```
public static void main(String[] arguments)
{
        System.out.println("hi");
}
```

Adding Methods

public static void ***NAME***() {

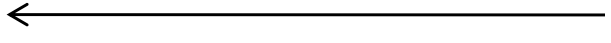
    STATEMENTS

}

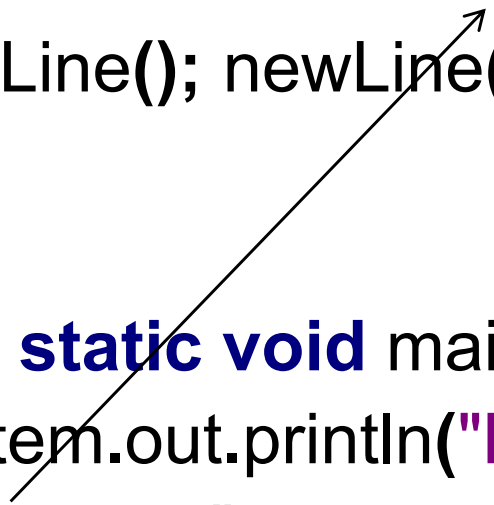
To call a method:

```
NAME();
```

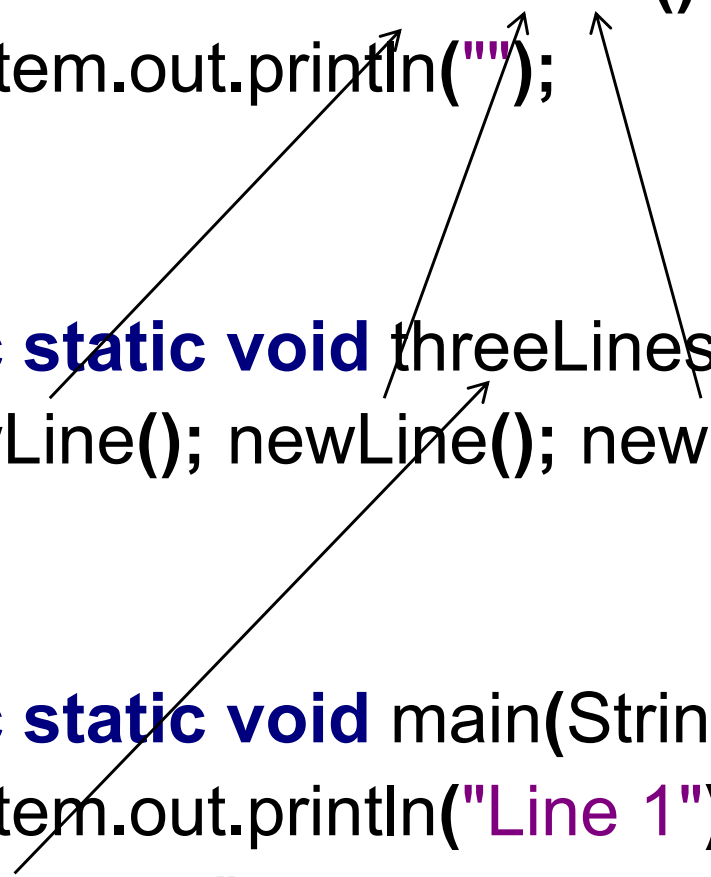
```
class NewLine {
    public static void newLine() {
        System.out.println("");
    }

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

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



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

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

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



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

    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 ***NAME***(***TYPE NAME***) {

    STATEMENTS

}

To call:

```
NAME(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");
        printSquare(5.5);
    }
}
```

What's wrong here?

Multiple Parameters

[…] ***NAME***(***TYPE NAME, TYPE NAME***) {

    STATEMENTS

}

To call:

```
NAME(arg1, arg2);
```

```
class Multiply {
    public static void times (double a, double b) {
        System.out.println(a * b);
    }

    public static void main(String[] arguments) {
        times (2, 2);
        times (3, 4);
    }
}
```

Return Values

public static ***TYPE*** ***NAME***() {

    STATEMENTS

    return ***EXPRESSION;***

}

`void` means “no type”

```
class Square4 {
    public static double square(double x) {
        return x*x;
    }

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

Mathematical Functions

```
Math.sin(x)

Math.cos(Math.PI / 2)

Math.pow(2, 3)

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

Conversion by method

int to String:

```
String five = 5; // ERROR!
String five = Integer.toString (5);
String five = “” + 5;  // five = “5”
```

String to int:

```
int foo = “18”; // ERROR!
int foo = Integer.parseInt (“18”);
```