Introduction to Java
A Farewell to Karel
Welcome to Java
But First...

A Brief History of Digital Computers
Programming in the 1940s

Electrical Device
High-Level Languages
Programming in the 1950s

DO 5 I = 1, 25

Source Deck

Compiler

Program Deck

Computer
Programming in the 1950s

DO 5 I = 1, 25

Source Deck

Compiler

Program Deck

Computer
Programming Now (ish)

Source Code

Compiler

Machine Code

move();
turnLeft();

11011100
10111110
Hey! I wrote a program that can draw stick figures!

That's great! I wrote a program that makes speech bubbles!
move();
turnLeft();

Compiler

Object File

11011
10111

Object File

11011
10111

Linker

Machine Code

11011
10111

Computer
move();
turnLeft();

Source Code

Compiler

Object File

Linker

Object File

Machine Code

Computer
move();
turnLeft();
Time-Out For Announcements
Section Assignments

- Section assignments given out on Tuesday; you can submit assignments once you have an SL assigned.
  - Didn't sign up? Signups reopen on Tuesday.
- Section Handout 1 released.
  - Recommendation: review the handout and think about the problem before attending section.
  - Section problems are not collected or graded; you'll go over them in section.
Announcements

• Programming Assignment #1 Out:
  • Karel the Robot: Due Friday, January 17 at 3:15 PM.
    - Suggestion: Try to have a working solution to all the Karel problems by Wednesday. That gives you two buffer days to do final testing and cleanup.
  • Email: Due Sunday, January 19 at 11:59PM.
    - Please wait until you get your section assignments before writing these emails – we'd like you to introduce yourself to your SL as well!
Getting Help

- **It's normal to ask for help in CS106A!**
- LaIR hours start tonight! 6PM – Midnight, Sunday through Thursday.
- Keith's Office Hours:
  - Tuesday, 10:15AM – 12:15PM in Gates 505.
  - Wednesday, 4:30PM – 6:30PM in Gates 505.
- QuestionHut (link on the CS106A website)
  - Q&A site for CS106A.
  - Keith and Vikas frequently look over it, and you can answer questions as well!
Let's See Some Java!
public class Add2Integers extends ConsoleProgram {
    public void run() {
        println("This program adds two numbers.");
        int n1 = readInt("Enter n1: ");
        int n2 = readInt("Enter n2: ");
        int total = n1 + n2;
        println("The total is "+ total + ".");
    }
}

17 25 42
Variables

• A **variable** is a location where a program can store information for later use.
Variables

• A **variable** is a location where a program can store information for later use.
Variables

• A **variable** is a location where a program can store information for later use.

• Each variable has three pieces of information associated with it:
Variables

• A **variable** is a location where a program can store information for later use.

• Each variable has three pieces of information associated with it:
  • **Name**: What is the variable called?
Variables

- A **variable** is a location where a program can store information for later use.

  ```java
  int numVoters
  ```

- Each variable has three pieces of information associated with it:
  - **Name**: What is the variable called?
Variables

• A **variable** is a location where a program can store information for later use.

• Each variable has three pieces of information associated with it:
  • **Name**: What is the variable called?
  • **Type**: What sorts of things can you store in the variable?

```
int numVoters
```
A **variable** is a location where a program can store information for later use.

```
int numVoters
```

Each variable has three pieces of information associated with it:

- **Name**: What is the variable called?
- **Type**: What sorts of things can you store in the variable?
Variables

• A **variable** is a location where a program can store information for later use.

```c
int numVoters
```

• Each variable has three pieces of information associated with it:
  
  • **Name**: What is the variable called?
  
  • **Type**: What sorts of things can you store in the variable?
  
  • **Value**: What value does the variable have at any particular moment in time?
Variables

- A **variable** is a location where a program can store information for later use.

```
137 int numVoters
```

- Each variable has three pieces of information associated with it:
  - **Name**: What is the variable called?
  - **Type**: What sorts of things can you store in the variable?
  - **Value**: What value does the variable have at any particular moment in time?
Variables

A variable is a location where a program can store information for later use.

Each variable has three pieces of information associated with it:

- **Name**: What is the variable called?
- **Type**: What sorts of things can you store in the variable?
- **Value**: What value does the variable have at any particular moment in time?
Variable Names

- Legal names for variables begin with a letter or an underscore (_).
- Consist of letters, numbers, and underscores, and aren't one of Java's reserved words.

Examples:
- x
- 7thHorcrux
- Harry Potter
- noOrdinaryRabbit
- lots_of underscores
- w
- LOUD_AND_PROUD
- that'sACoolName
- true
- C_19_H_14_O_5_S
Variable Names

• Legal names for variables
  • begin with a letter or an underscore (_)

- x
- 7thHorcrux
- Harry Potter
- noOrdinaryRabbit
- lots_of_underscores

- w
- LOUD_AND_PROUD
- that'sACoolName
- true
- C_19_H_14_O_5_S
Variable Names

• Legal names for variables
  • begin with a letter or an underscore (_)

x
7thHorcrux
Harry Potter
noOrdinaryRabbit
lots_of_underscores

w
LOUD_AND_PROUD
that'sACcoolName
true
C_19_H_14_O_5_S
Variable Names

- Legal names for variables
  - begin with a letter or an underscore (_)
  - consist of letters, numbers, and underscores, and aren't one of Java's reserved words.

<table>
<thead>
<tr>
<th>x</th>
<th>w</th>
</tr>
</thead>
<tbody>
<tr>
<td>7thHorcrux</td>
<td>LOUD_AND_PROUD</td>
</tr>
<tr>
<td>Harry Potter</td>
<td>that'sACoolName</td>
</tr>
<tr>
<td>noOrdinaryRabbit</td>
<td>true</td>
</tr>
<tr>
<td>lots_of_underscores</td>
<td>C_19_H_14_O_5_S</td>
</tr>
</tbody>
</table>
Variable Names

- Legal names for variables
  - begin with a letter or an underscore (_)
  - consist of letters, numbers, and underscores,

  x
  7thHorcrux
  Harry-Potter
  noOrdinaryRabbit
  lots_of_underscores

  w
  LOUD_AND_PROUD
  that'sACoolName
  true
  C_19_H_14_O_5_S
Variable Names

• Legal names for variables
  • begin with a letter or an underscore (_)
  • consist of letters, numbers, and underscores, and
  • aren't one of Java's reserved words.

x
7thHorcrux
Harry---Potter
noOrdinaryRabbit
lots_of Underscores

w
LOUD_AND_PROUD
that'sACoolName
true
C_19_H_14_O_5_S
Variable Names

• Legal names for variables
  • begin with a letter or an underscore (_)
  • consist of letters, numbers, and underscores, and
  • aren't one of Java's reserved words.

x
7thHorcrux
Harry-Potter
noOrdinaryRabbit
lots_of_underscores
w
LOUD_AND_PROUD
that'sACoolName
true
C_19_H_14_O_5_S
Variable Names

• Legal names for variables
  • begin with a letter or an underscore (_)
  • consist of letters, numbers, and underscores, and
  • aren't one of Java's reserved words.

    x
    w
    LOUD_AND_PROUD

    noOrdinaryRabbit
    lots_of_underscores
    C_19_H_14_O_5_S
Variable Naming Conventions

• You are free to name variables as you see fit, but there are some standard conventions.

• Names are often written in **lower camel case:**

  capitalizeAllWordsButTheFirst
Variable Naming Conventions

- You are free to name variables as you see fit, but there are some standard conventions.
- Names are often written in **lower camel case**: `capitalizeAllWordsButTheFirst`
Variable Naming Conventions

- You are free to name variables as you see fit, but there are some standard conventions.
- Names are often written in **lower camel case**: capitalizeAllWordsButTheFirst

```python
capitalizeAllWordsButTheFirst
```
Variable Naming Conventions

• You are free to name variables as you see fit, but
  there are some standard conventions.
• Names are often written in lower camel case:
  capitalizeAllWordsButTheFirst
• Choose names that describe what the variable does.
  • If it's a number of voters, call it numberOfVoters,
    numVoters, voters, etc.
  • Don't call it x, volumeControl, or severusSnape
Types

• The type of a variable determines what can be stored in it.

• Java has several primitive types that it knows how to understand:
Types

- The **type** of a variable determines what can be stored in it.

- Java has several **primitive types** that it knows how to understand:
  - **int**: Integers.
Types

• The **type** of a variable determines what can be stored in it.

• Java has several **primitive types** that it knows how to understand:
  • **int**: Integers.
  • **double**: Real numbers.
Types

- The **type** of a variable determines what can be stored in it.
- Java has several **primitive types** that it knows how to understand:
  - **int**: Integers.
  - **double**: Real numbers.
Types

- The **type** of a variable determines what can be stored in it.
- Java has several **primitive types** that it knows how to understand:
  - **int**: Integers. *(counting)*
  - **double**: Real numbers.
Types

• The **type** of a variable determines what can be stored in it.

• Java has several **primitive types** that it knows how to understand:
  
  • **int**: Integers. (counting)
  
  • **double**: Real numbers. (measuring)
Types

- The **type** of a variable determines what can be stored in it.
- Java has several **primitive types** that it knows how to understand:
  - **int**: Integers. *(counting)*
  - **double**: Real numbers. *(measuring)*
  - **boolean**: Logical true and false.
Types

- The **type** of a variable determines what can be stored in it.

- Java has several **primitive types** that it knows how to understand:
  - **int**: Integers. *(counting)*
  - **double**: Real numbers. *(measuring)*
  - **boolean**: Logical true and false.
  - **char**: Characters and punctuation.
Values

137  \hspace{1cm} \text{int} \hspace{0.5cm} \text{numVotes}

0.97333 \hspace{1cm} \text{double} \hspace{0.5cm} \text{fractionVoting}

0.64110 \hspace{1cm} \text{double} \hspace{0.5cm} \text{fractionYes}
Declaring Variables
Declaring Variables

public void run() {

}
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
}
```
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
}
```
Declaring Variables

```
public void run() {
    double ourDouble = 2.71828;
}
```

The syntax for declaring a variable with an initial value is

```
type name = value;
```
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;
}
```
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;
}

2.71828
ourDouble

137
ourInt
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
}
```
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
}
```
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;

    Variables can be declared without an initial value:
    type name;
}
```
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
    anotherInt = 42;
}
```
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
    anotherInt = 42;
}
```
Declaring Variables

public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
    anotherInt = 42;

    An assignment statement has the form
    variable = value;

    This stores value in variable.
}
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;
    int anotherInt;
    anotherInt = 42;
    ourInt = 13;
}

2.71828
ourDouble

137
ourInt

42
anotherInt
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
    anotherInt = 42;

    ourInt = 13;
}
```
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
    anotherInt = 42;

    ourInt = 13;
}
```
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
    anotherInt = 42;

    ourInt = 13;
    ourInt = ourInt + 1;
}

Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
    anotherInt = 42;

    ourInt = 13;
    ourInt = ourInt + 1;
}
```
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
    anotherInt = 42;

    ourInt = 13;
    ourInt = ourInt + 1;
}
```
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
    anotherInt = 42;

    ourInt = 13;
    ourInt = ourInt + 1;

    anotherInt = ourInt;
}
```
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;
    int anotherInt;
    anotherInt = 42;
    ourInt = 13;
    ourInt = ourInt + 1;
    anotherInt = ourInt;
}

public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
    anotherInt = 42;

    ourInt = 13;
    ourInt = ourInt + 1;

    anotherInt = ourInt;
}

Declaring Variables

```
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
    anotherInt = 42;

    ourInt = 13;
    ourInt = ourInt + 1;

    anotherInt = ourInt;
    ourInt = 1258;
}
```
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;
    int anotherInt;
    anotherInt = 42;
    ourInt = 13;
    ourInt = ourInt + 1;
    anotherInt = ourInt;
    ourInt = 1258;
}
```
Declaring Variables

```java
public void run() {
    double ourDouble = 2.71828;
    int ourInt = 137;

    int anotherInt;
    anotherInt = 42;

    ourInt = 13;
    ourInt = ourInt + 1;

    anotherInt = ourInt;
    ourInt = 1258;
}
```