# Chapter 1: Introduction

## CS 121

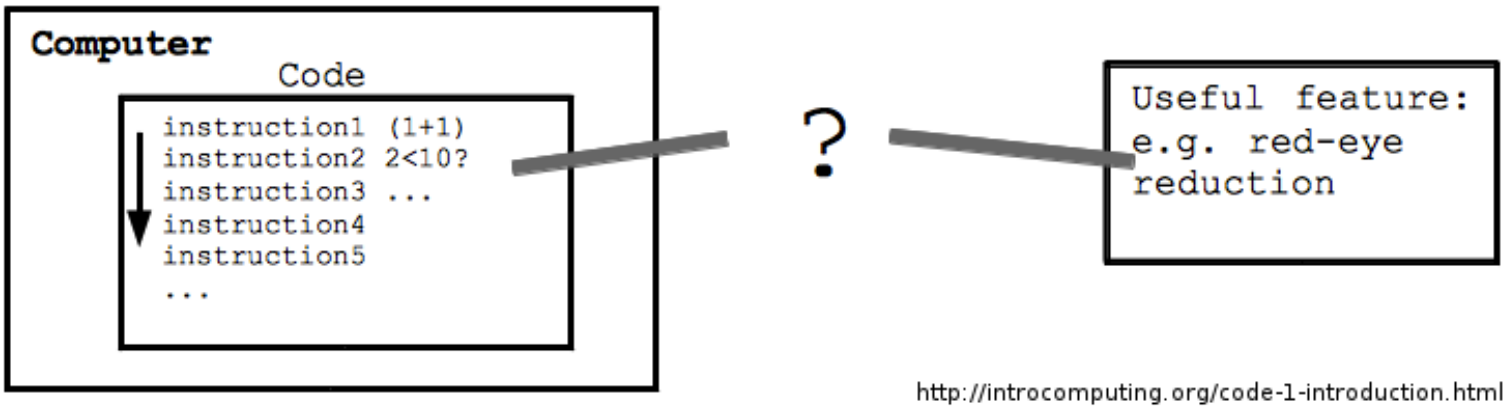
Department of Computer Science
College of Engineering
Boise State University

January 20, 2015

# Topics

- What is Computer Science?
- Problem solving techniques
- Program development in Java

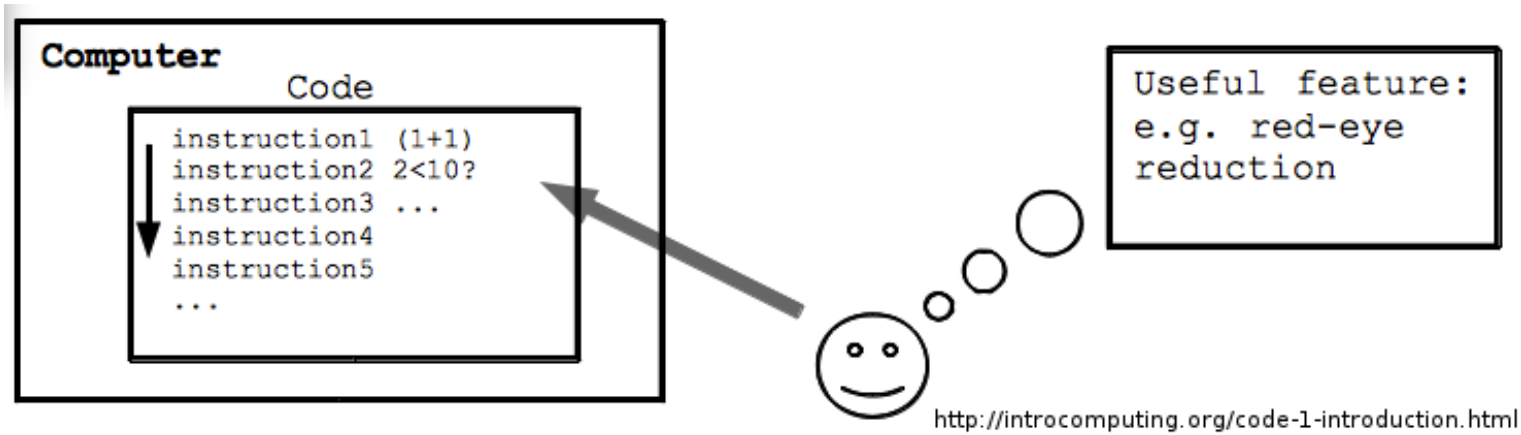
# How does a computer work?




http://introcomputing.org/code-1-introduction.html

- Computer is driven by code.
- Code is made of simple, mechanical instructions.
- Computer runs series of instructions.
- Machine simply follows directions, so how does it do so much??

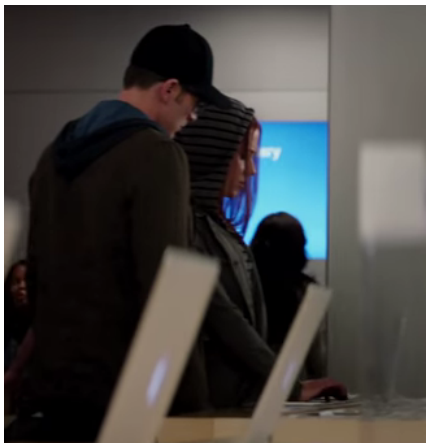
# Humans to the rescue




- People come up with cool ideas (requirements),
- think through the solutions (design),
- break them down into simple instructions for the computer (algorithms),
- write code for the computer (implementation), and
- test them to make sure the computer is behaving properly (testing).

# Computer Science - not just about writing code.

- More to computer science than writing code.
- Teaches people how to break down and solve problems algorithmically.
- Gives people the tools to solve problems in *all* human endeavors.
  - Farming, finance, entertainment, engineering, art, music etc.
- Every company will soon become a software company!
- Even superheroes need computer science.



# What is Computer Science?

- Computer Science is the art and science behind creating and running software.
- Major areas in computer science:
  - How to solve problems and how to organize data?
    - Algorithms and Data Structures
  - How to express a solution in code? Programming Languages
    - Programming Languages
  - How to design, manage and maintain the systems to run the code?
    - Systems: Computer Architecture, Operating Systems, Networks, Databases, Distributed Systems (Cloud Computing, Servers, Web), etc.
- Software engineering is the application of tools and techniques from all three areas above to large projects.

# Program Development

- Program development (or implementation) can be broken into the following tasks.
  - Programming – writing the program.
  - Compilation – translating the program into a form the computer can execute (machine-language).
  - Execution – running the program.
  - Debugging – investigating and fixing various types of errors that occur.
- The set of tools you choose to use in this process make up your development environment.

# Program Development

- Programming – writing the program.
- Compilation – translating the program into a form the computer can execute (machine-language).
- Execution – running the program.
- Debugging – investigating and fixing various types of errors that occur.

# Programming Languages

- A program controls a computer. Programs are also called *applications*.
- A programming language specifies the words and symbols we can use to write a program.
- A programming language employs a set of rules that dictate how the words and symbols can be put together to form valid program statements.

# High-Level Programming Language

- We write programs in high-level languages.
  - Understandable by humans.
  - Machine independent (runs on different computer models).
  - Needs to be translated by a compiler.
- In this class, we will write programs in a high-level language called Java.

# Program Development in Java

- There are many high-level computer programming languages.
  - Java
  - C/C++
  - C#
  - Python
  - JavaScript
- So, why Java?
  - Concepts you learn can be applied to other programming languages.
  - One of the world's most popular programming languages.
  - At the core of many applications you use every day.

# Program Development in Java

- Java is freely available.
- The Java Development Kit (JDK) is for developing Java programs.
- The Java Runtime Environment (JRE) is for running Java programs.
- Latest major version is Java 1.8.

# Programming

- Every language has its own set of syntax rules and semantics.
  - Syntax rules – how we can put together symbols, reserved words, and identifiers. (`c = a + b`)
  - Semantics – rules that define what a program statement means. (`c = a + b` means: add the variables `a` and `b` and store the result in `c`)
  - Think alphabet and grammar.
- A program will always do what we tell it to do, not what we **meant** to tell it to do.

# Basic Programming in Java

- Example: Lincoln.java
- Example: Lincoln2.java and Lincoln3.java

# Let's break it down.

- Class header
- Class body
- Comments
- Method header
- Method body
- Identifiers
- White space

# In-Class Exercise

What do you think the following code does?

```
public class HelloClass
{
    public static void main(String[] args)
    {
        // Let's say hello.
        System.out.println("Hello CS 121!"));
    }
}
```

1. Prints "Hello CS 121!" to the screen and terminates.
2. Prints "Hello CS 121!" to the screen and keeps running.
3. Prints "Let's say hello. Hello CS 121!" to the screen and terminates.
4. Does nothing useful. There is an error.

# Program Development

- Programming – writing the program.
- Compilation – translating the program into a form the computer can execute (machine-language).
- Execution – running the program.
- Debugging – investigating and fixing various types of errors that occur.

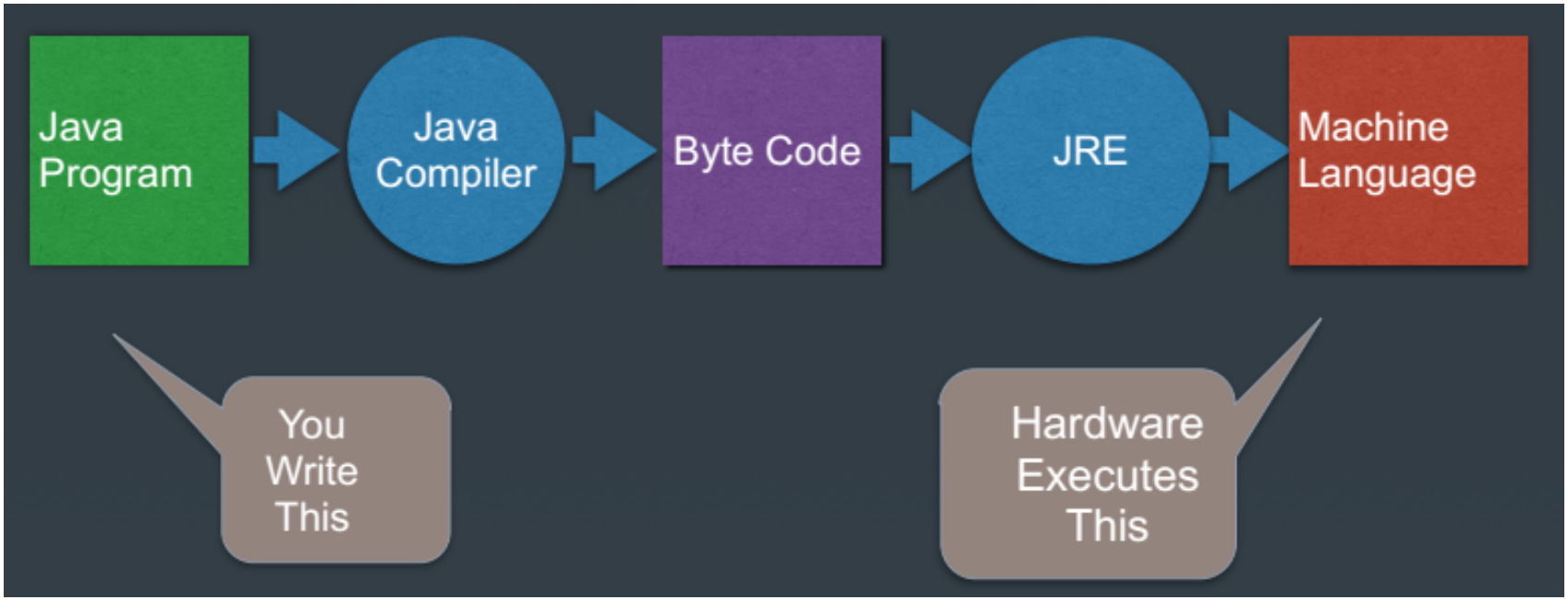
# Machine Language

- Computer hardware can only execute a machine language program.
  - Tells hardware what to do.
  - Sequence of binary numbers (e.g. 0001 0010 0011 0100...)
  - Machine langauge is specific to hardware brand/model.
  - Difficult for most humans to compose, edit, and understand.
- We don't write programs in machine language, so how do we tell it what to do?

# Compilation

- A program must *translated* into machine language before it can be executed.
- A compiler translates a high-level language into another programming language.
- A compiler is just a computer program, a programmer's tool.
- The Java compiler is included with the Java Development Kit (JDK).

# The Big Picture



# Program Development

- Programming – writing the program.
- Compilation – translating the program into a form the computer can execute (machine-language).
- Execution – running the program.
- Debugging – investigating and fixing various types of errors that occur.

# Execution

- After a program is compiled, you may run it from the command-line.

# Program Development

- Programming – writing the program.
- Compilation – translating the program into a form the computer can execute (machine-language).
- Execution – running the program.
- Debugging – investigating and fixing various types of errors that occur.

# Debugging

- Where you will probably spend most of your time.
- Three major types of errors.
  - compile-time: Incorrect syntax or other basic problems
  - run-time: A problem during program excecution that causes it to terminate abnormally. E.g. divide by zero etc.
  - logical: The program runs but produces incorrect results. E.g. Using the wrong formula, producing the wrong output etc.

# In-class Exercise

**Ex 1.6**. Categorize each of the following as a compile-time error, run-time error, or logical error.

- multiplying two numbers when you meant to add them
- dividing by zero
- forgetting a semicolon at the end of a code statement
- spelling a word wrong in the output
- producing inaccurate results
- typing a { when you should have typed a (

# Example

- Enough talk...
- Let's just break it and see what happens.

# Exercises

- Activate your Piazza account.
- Read Chapter 1. Focus on sections 1.1, 1.3 and 1.4 and skim the rest.
- **Recommended Homework**
  - Exercises: EX 1.1-1.6.
  - Projects: PP 1.4, 1.7.
- Browse Chapter 2.