Introduction to EECS1020

Instructor: Steven Castellucci

Website

www.eecs.yorku.ca/course/1020/

Syllabus

- Required Textbook
  - Java By Abstraction: A Client-View Approach, by H. Roumani. Publisher: Pearson Ed.
- Evaluation

◦ Test 1:	15%
◦ Test 2:	15%
◦ Test 3:	15%
◦ Test 4:	15%
◦ Test 5:	15%
◦ Final Exam:	25%

Goals of 1020

- Programming fundamentals
- Object-oriented concepts
- Problem solving

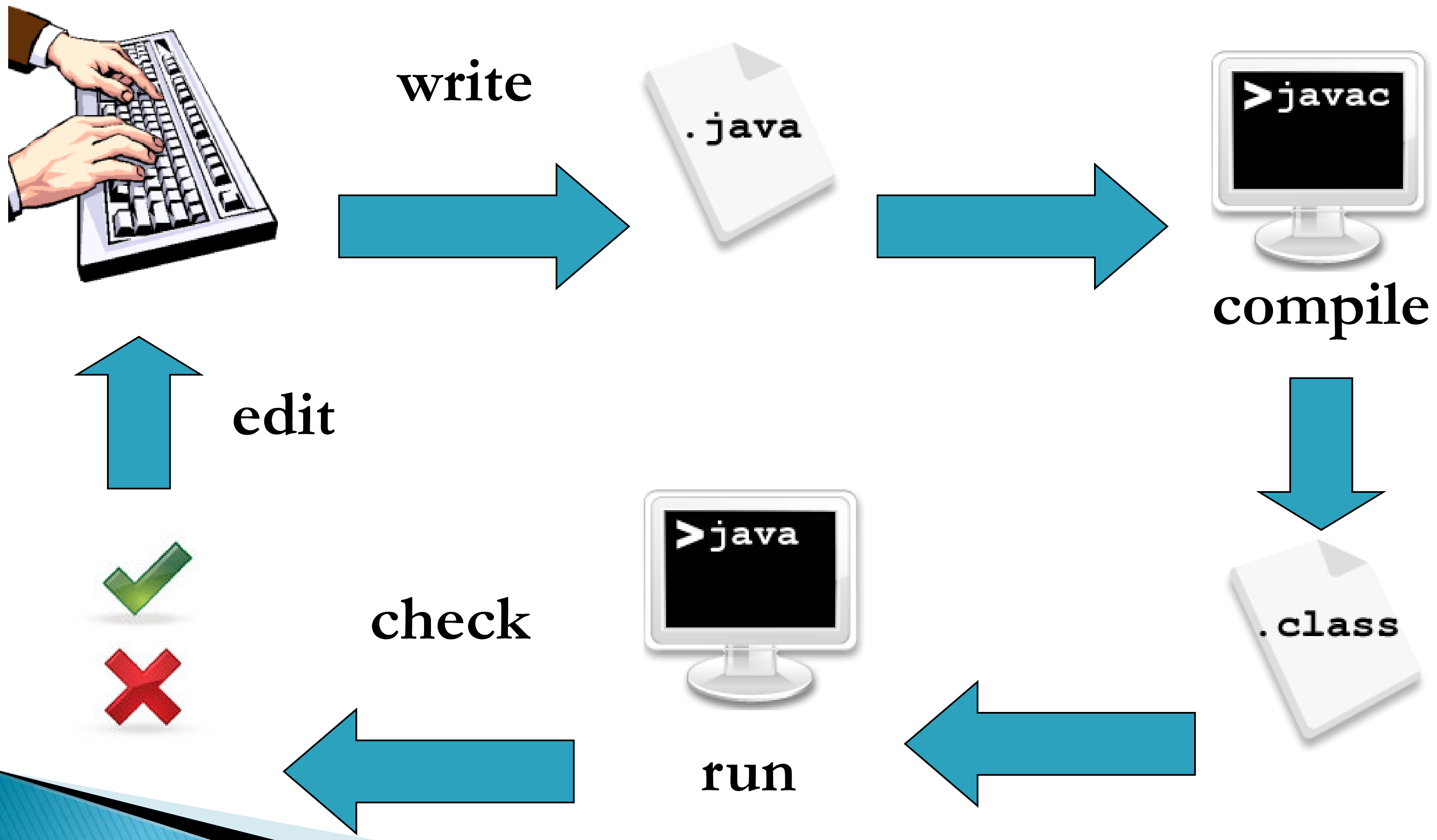
Why Program?

- Computers can
  - Perform billions of mathematical and logical operations every second
  - Operate continuously without error
  - Allow communications over vast distances
  - Facilitate the sharing of ideas, information, and knowledge

Why Program?

- But, computers are… dumb!
- Computers can only accomplish what their program tells them to do
- Programmers help make computers the valuable resource that they are

How to Program



How to Program

- Write code in a computer language (remember to save the file)
- Use the language's compiler to convert your code to machine-readable code
- Run your program
- Compare actual result to expected one
- Edit your code as necessary, and repeat

Why Java?

- Syntax is similar to other languages
- Platform independent
- Widely used in industry

- Uses:
  - Server-side (Gmail backend)
  - Client-side (jEdit, Eclipse)
  - Mobile devices (Android*)
  - Consumer devices (Blu-ray players)

*Android programs use Java syntax and similar libraries.

Machine Language

- Why not just program in machine language?
- Machine languages are
  - Machine-dependant
  - Complex and verbose
  - Difficult to understand large programs
- Compilers abstract (i.e., remove) the complexities of machine language
- Programming languages simplify design

Object-Oriented Programming

- Object-oriented languages (e.g., Java) encapsulate (i.e., represent) real-world concepts as “objects”
- Objects (and methods to operate on them) are defined in entities called “classes”
- Java includes a library of predefined classes defined in an Application Programming Interface (API)

Abstraction Levels

- Abstraction allows a programmer to focus on a single responsibility
- Focus is either “high–level” or “low–level”
  - High–level: simple, general (e.g., print(5 + 3))
  - Low–level: complex, specific (e.g., store the integer value 5 in memory register $1, store the integer value 3 in memory register $2, add the values in $1 and $2 and place their sum in $3, print the contents of $3 to the screen)

Design Methodologies

- Top-down (high-level to low-level)
  - Start with general requirements
  - Divide into specific responsibilities
  - Implement components for each
- Bottom-up (low-level to high-level)
  - Identify the primitive operations required
  - Implement modules to perform such tasks
  - Facilitate collaboration between the modules to meet the required specifications

Focus of Chapter One and Two

- Chapter One
  - Low-level
  - Java syntax
  - Data types and ranges
- Chapter Two
  - High-level
  - Abstraction
  - Client-Implementer delegation

Questions?