Introduction

CMSC 202

Spring 2012

Instructors & Lecture Sections

- Mr. Ryan Bergeron
  - Section 04, 07, 10
- Mr. Wes Griffin
  - Section 01
- Ms. Susan Mitchell
  - Section 16
- Mr. Michael Rushe
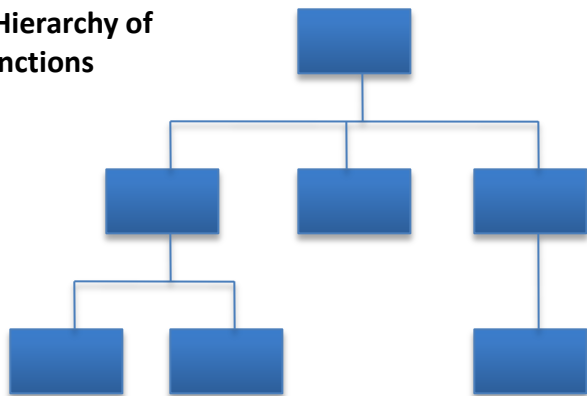 - Section 13

What is CMSC 202?

- An introduction to object-oriented programming (OOP) and object-oriented design (OOD)
  - Uses the Java programming language
  - Uses the Eclipse integrated development environment (IDE)
- Strong emphasis on proper program design
- Course website:
  www.cs.umbc.edu/courses/undergraduate/202/spring12

Procedural vs. OO Programming

Procedural

- Modular units: functions
- Program structure: hierarchical
- Data and operations **are not** bound to each other
- Examples:
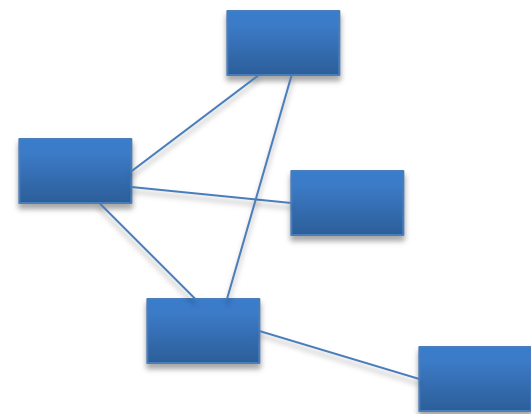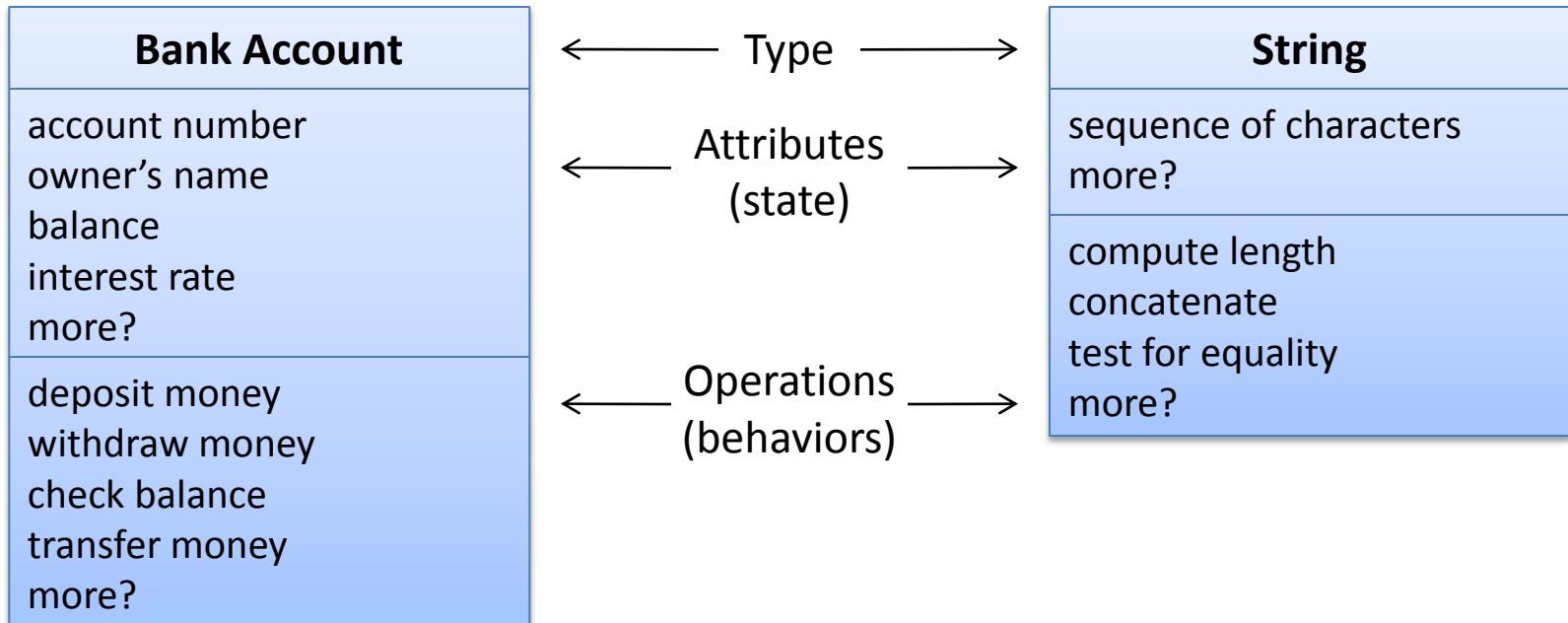 - C, Pascal, Basic, Python

Object-Oriented (OO)

- Modular units: objects
- Program structure: a graph
- Data and operations **are** bound to each other
- Examples:
  - Java, C++, Ruby

A Hierarchy of Functions

A Collection of Objects

What's an Object?

- Must first define a **class**
  - A <u>data type</u> containing:
    - Attributes – make up the object's "state"
    - Operations – define the object's "behaviors"

Bank Account
account number owner's name balance interest rate more?
deposit money withdraw money check balance transfer money more?

←——— Type ———→

←——— Attributes (state) ———→

←——— Operations (behaviors) ———→

String
sequence of characters more?
compute length concatenate test for equality more?

5

So, an Object is...

- A particular <u>instance</u> of a class

Bergeron's Account

12-345-6
Ryan Bergeron
$1,250.86
1.5%

Frey's Account

65-432-1
Dennis Frey
$5.50
2.7%

Mitchell's Account

43-261-5
Susan Mitchell
$825.50
2.5%

For any of these accounts, one can...
- Deposit money
- Withdraw money
- Check the balance
- Transfer money

Why Java for 202?

- Popular modern OO language

- Wide industry usage

- Used in many types of applications

- Desirable features
  - Object-oriented
  - Portability (cross-platform)
  - Easy handling of dynamic variables
  - Garbage collection
  - Built-in GUI libraries

Java History

- Created by **Sun Microsystems** team led by **James Gosling** (1991)

- Originally designed for programming home appliances
  - Difficult task because appliances are controlled by a wide variety of computer processors
  - Writing a compiler (translation program) for each type of appliance processor would have been very costly
  - Solution: **two-step translation process**
    - Compile, then
    - Interpret

Interpreters, Compilers, and the JVM

Interpreted Languages (e.g. JavaScript, Perl, Ruby)
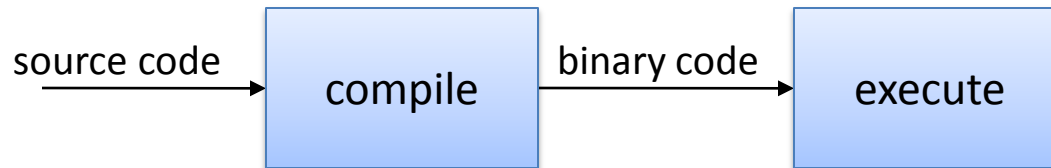
source code → interpret
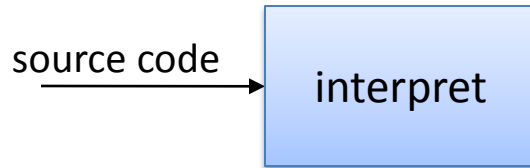
Interpreter translates code into binary and executes it

Small, easy to write

Interpreter is unique to each platform

Compiled Languages (e.g. C, C++)

source code → compile → binary code → execute

Compiler is unique to each platform

Java

source code → compile → *bytecode* → interpret

Java Virtual Machine (JVM)
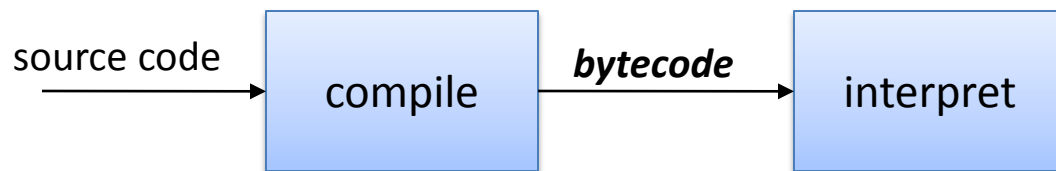
Bytecode is <u>platform independent</u>

JVM is unique to each platform

Compiling and Running C/C++

Project Library
for Linux

Linux
binary

Linux
executable

C/C++
Code

Linux C/C++ compiler

Linux C/C++ linker

Windows C/C++ compiler

Project Library
for Windows

Windows
binary

Windows
executable

Windows C/C++ linker
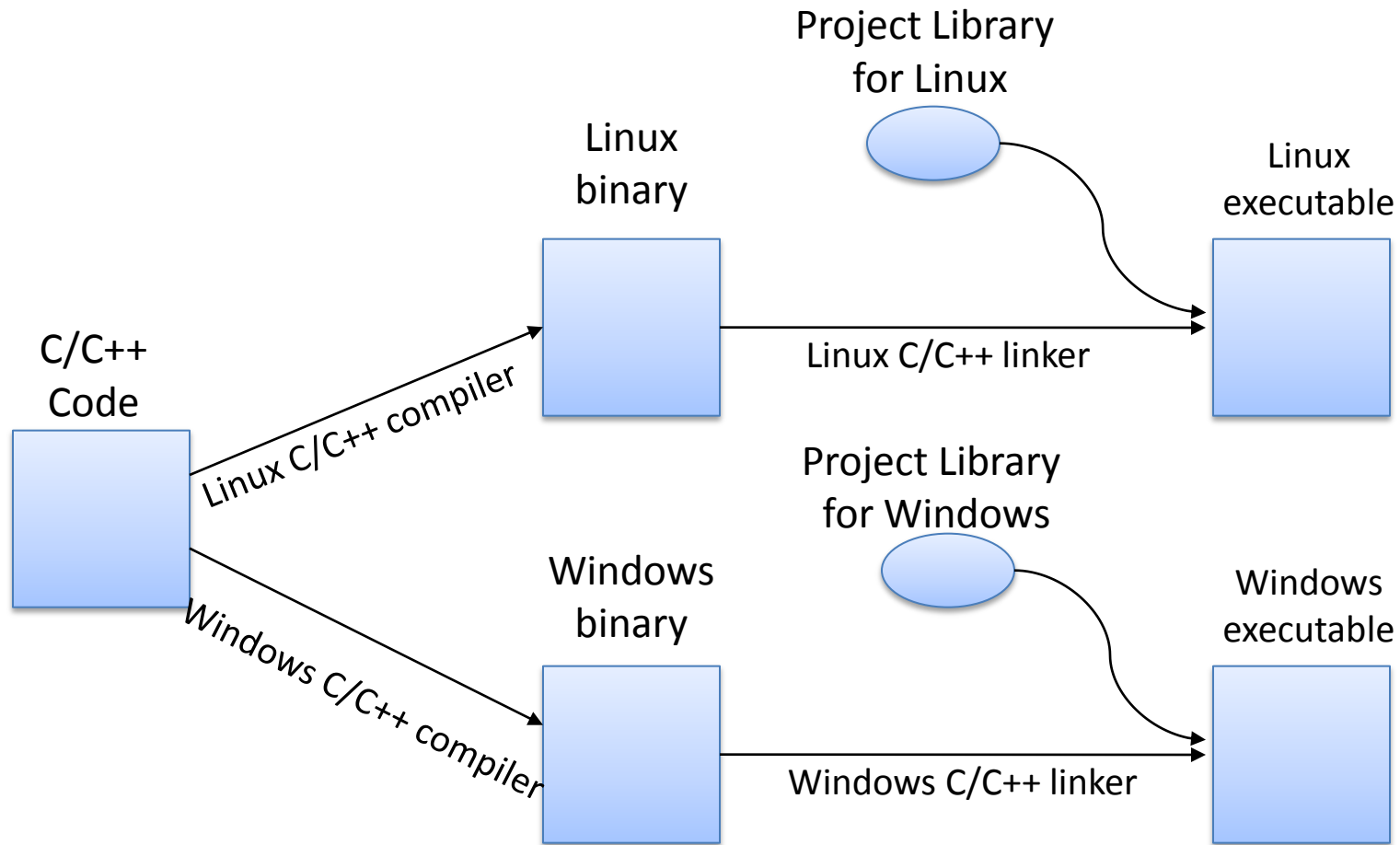
Compiling and Running Java

Java
Code

Java
Bytecode

JRE for Linux

javac Hello.java

Java compiler

java Hello

Java interpreter (JVM)
translates bytecode to
machine code in JRE

Hello.java

Hello.class

java Hello

JRE for
Windows

Java Terminology

- Java acronyms are plentiful and confusing. Here are the basics.

  - JVM – Java Virtual Machine
    - Translates Java bytecode to machine code
  - API – Application Programming Interface
    - The classes/methods/constants provided by libraries
  - JRE – Java Runtime Environment
    - The JVM and the Java API together
  - JDK (formerly SDK) – Java Development Kit
    - JRE + tools (compiler, debugger) for developing Java applications
  - Java SE – Java Platform, Standard Edition
    - The given edition of the JRE – standard being the most common
    - There are other versions that are tailored toward mobile devices and web environments

- To learn more about JDK, JRE, etc, visit:
  - http://www.oracle.com/technetwork/java/javase/tech/index.html

Java SE Versions

- Current version of Java: Java 7, also known as Java 1.7 or Java 1.7.0

- Previous version: Java 6, also known as Java 1.6, Java 1.6.0 or "Java 2 SE Version 6"
  - This is the version running on GL servers

- To learn more about Java version naming, see: http://java.sun.com/javase/namechange.html

Python vs. Java

- ## Python

```
print "Hello, world"
quotient = 3 / 4
if quotient == 0:
    print "3/4 == 0",
    print "in Python"
else:
    print "3/4 != 0"
```

Things to note:
- Everything has to be in some class
- We need a "main()"
- Statements end with ";"
- Variables must be declared
- "if/else" syntax different
- Statement blocks demarcated by "{...}"
- Comments are different
- Much that is similar

- ## Java

```
public class Hello {
  public static void main(String[] args) {
    int quotient;
    System.out.println("Hello, world");
    quotient = 3 / 4;
    if (quotient == 0) {
      System.out.print("3/4 == 0");
      System.out.println(" in Java");
    } else {
      System.out.println("3/4 != 0");
    }
  }
}
```

The Eclipse IDE

- An integrated development environment (IDE) for writing Java programs. Contains (minimally):
  - Editor
  - Debugger
  - Java compiler
  - Java JVM
- Free (open source) download for Windows/Linux/Mac
  - See course "Resources" page on the CMSC 202 website
- Available in all OIT labs around campus
  - We'll show you more in Lab 1

Eclipse IDE Screenshot