Object Oriented Concepts and Programming (CSC244)

By

Dr. Tabbasum Naz

tabbasum.naz@ciitlahore.edu.pk

Course Outline

Course Title	•Object Oriented Concepts and Programming
Course Code	CSC244
Credit Hours	4(3,1)
Semester	Spring 2012
Resource Person	Dr. Tabbasum Naz
Contact Hours (Theory)	3 hours per week
Contact Hours (Lab)	3 hours per week
Office Hours	Shall be communicated later
Email	tabbasum.naz@ciitlahore.edu.pk

Course Outline

GENERAL OVERVIEW

- This course provides an introduction to the concepts and methodology of Object-Oriented Programming with Java as an illustration language.
- Course also help students to implement object-oriented concepts keeping in mind the real-world problems.

COURSE OBJECTIVES

- Understand major concepts of object-oriented programming
- Knowledge and skills in OO design and program development
- Experience in Java programming and program development within an integrated development environment
- Certain skills in using graphical user interface
- Topics covered in this course provide a foundation for more advanced courses in Computer Science

Course Outline

TEXT BOOK

"Java How to Program", by Deitel & Deitel, 7th Edition

RECOMMENDED BOOKS

- "Java 2: The Complete Reference", by Patrick Naughton and Herbert Schildt
- "Thinking in Java" by Bruce Eckel, Prentice Hall, 4th Edition, 2006
- "Beginning Java 2" by Ivor Horton

Course Assessment

Theory Part

- Sessional-I Exam 10%
- Sessional-II Exam 15%
- Final Exam 50%
- Quiz (3-6 per semester) 15%
- Assignments (3-6 per semester) 10%

- Quizzes may be unannounced. There is no make-up for missed quiz.

Practical / Lab Part

- Performance or experiments 25%
- Sessional-I Exam 10%
- Sessional-II Exam 15%
- Final Exam 50%

- The minimum pass marks for each course shall be 50%. Students obtaining less than 50% marks in any course shall be deemed to have failed in that course.

Course Assessment

Grades	Letter Grade	Credit Points	Percentage Marks
A	(Excellent)	4.0	90 and above
A-		3.7	85-89
B+		3.3	80-84
B	(Good)	3.0	75-79
B-		2.7	70-74
C+		2.3	65-69
C	(Average)	2.0	60-64
C-		1.7	55-59
D	(Minimum passing)	1.3	50-54
F	(Failing)	0.0	Less than 50

Course Outline

Weeks	Topic of Lecture	Reading Assignment
Week 1	▪ Course Overview ▪ Introduction to Java, History ▪ The Java Development Environment	Chapter 1& 2
Week 2 & 3	▪ Object Oriented Concepts in Java, ▪ Classes, Objects, Methods ▪ Constructors ▪ Difference between OOP and procedural languages	Chapter 3
Week 4	▪ Control Statements if, if-else, while, do-while, switch	Chapter 4 & 5
Week 5	▪ Methods ▪ Unified Modeling Language ▪ State Diagram ▪ Use Case Diagram ▪ Class Diagram ▪ Activity Diagram	Chapter 6 + Handouts
Week 6	▪ Arrays	Chapter 7
Week 7	▪ A deeper look into Java Classes and Objects, ▪ Encapsulation and data hiding ▪ Data abstraction	Chapter 8 + Handouts

Course Outline (Contd)

Week 8 & 9	▪ Object Oriented Programming: Inheritance	Chapter 9 + Handouts
Week 10	▪ Object Oriented Programming: Polymorphism	Chapter 10
Week 11 & 12	▪ GUI components and Graphics	Chapter 11 & 12
Week 13	▪ Exception Handling	Chapter 13
Week 14	▪ Introduction to Java Applets	Chapter 20
Week 15 & 16	▪ Accessing Database with JDBC	Chapter 25 & Handouts

Attendance Policy

- A Minimum of 80% of attendance is required for sitting in the Final/Terminal examination.

- Any student having less than 80% attendance will receive “F” Grade regardless of his/her performance in previous exams, quizzes and assignments etc.

Machine, Assembly and High Level Languages

- **Machine Language**
  - (1,0)
  - Cumber some for humans
- **Assembly Language**
  - English like abbreviations
  - Need translator called assembler
  - Example: load basepay, add overpay, store gross pay
- **High Level Languages**
  - Need Compilers to convert HLL to machine language
  - Everyday English like statements
  - Most preferable
  - C,C++, .NET languages (e.g., Visual Basic, Visual C++ and C#), **JAVA (the most widely used)**

C, C++ and Java History

C →	C++ →	Java
1972	Early 1980s	1995
Dennis Ritchie (Bell Laboratories)	Bjarne Stroustrup(Bell Laboratories)	James Gosling (Sun Microsystems)
UNIX OS development language	• Capabilities of Object Oriented Programming • Hybrid Language (C/Object Oriented Style)	• General purpose • Class based Object Oriented • Write once, run anywhere

Introduction to Object Oriented Programming

- Java: Today's most popular software development language
- Developed by Sun Microsystems
- Implementation at **java.sun.com/j2se**
- Object Oriented Programming
- Portable (computer programs written in the Java language must run similarly on any hardware/operating-system platform).

JAVA

- James Gosling, Mike Sheridan, and Patrick Naughton initiated the Java language project in June 1991.
- Java was originally designed for interactive television, but it was too advanced for the digital cable television industry at the time.
- The language was initially called *Oak* after an oak tree that stood outside Gosling's office; it went by the name *Green* later, and was later renamed *Java*, from Java coffee, said to be consumed in large quantities by the language's creators.
- Sun Microsystems released the first public implementation as Java 1.0 in 1995. It promised "Write Once, Run Anywhere" (WORA), providing no-cost run-times on popular platforms.
- With the advent of *Java 2* , new versions had multiple configurations built for different types of platforms. For example, *J2EE* targeted enterprise applications and the greatly stripped-down version *J2ME* for mobile applications (Mobile Java). *J2SE* designated the Standard Edition.
- In 2006, for marketing purposes, Sun renamed new *J2* versions as *Java EE*, *Java ME*, and *Java SE*, respectively.
- In 2007, Sun made the Java's core code available under free software/open-source distribution terms.

Goals: Java Language

- It should be "simple, object-oriented and familiar".
- It should be "robust and secure".
- It should be "architecture-neutral and portable".
- It should execute with "high performance".
- It should be "interpreted, threaded, and dynamic".

Versions

- Major release versions of Java, along with their release dates:
- JDK 1.0 (January 23, 1996)
- JDK 1.1 (February 19, 1997)
- J2SE 1.2 (December 8, 1998)
- J2SE 1.3 (May 8, 2000)
- J2SE 1.4 (February 6, 2002)
- J2SE 5.0 (September 30, 2004)
- Java SE 6 (December 11, 2006)
- Java SE 7 (July 28, 2011)

Benefits of Java to Programming Community

- Support GUIs and multimedia capabilities such as graphics, images, animation, audio and video.
- Run on internet and communicate with other applications
- Applications can take advantage of the flexibility and performance improvements of multithreading
- Applications with richer file processing than is provided by C or C++.

Benefits of Java to Programming Community

- Not limited to the desktop or even to some local computer network, but can integrate Internet components and remote databases as well.

- Applications that can be written quickly and correctly in a manner that takes advantage of prebuilt software components.

- Easy access to a growing universe of reusable software components.

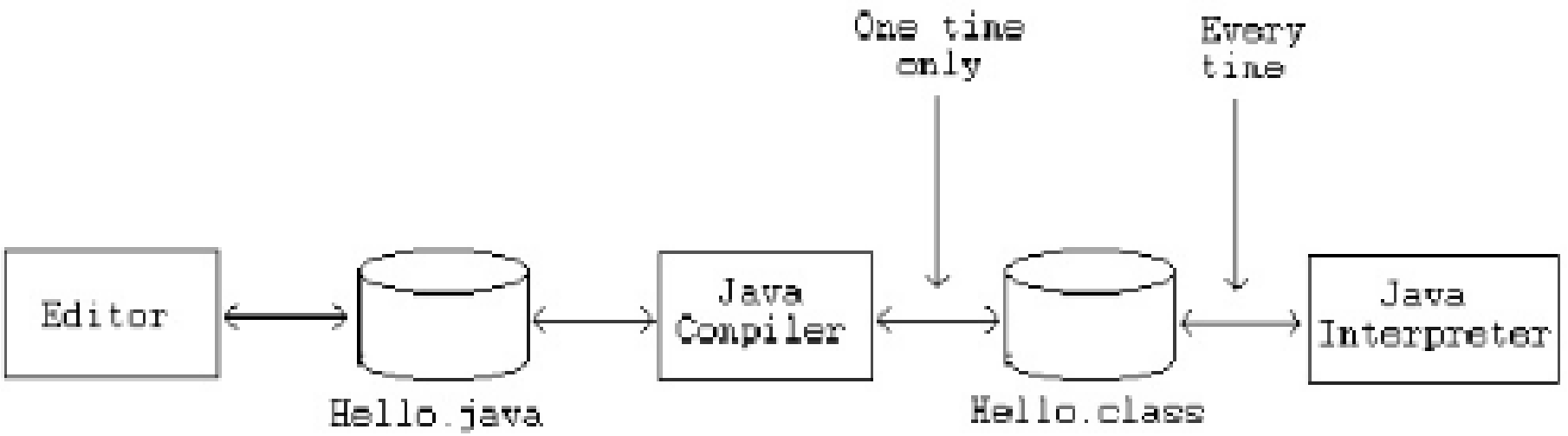
- Programmers want all these benefits in a truly portable manner, so that applications will run without modification on a variety of *platforms*

Motivation

- The original motivation for java
- The need for platform independent language that could be embedded with various consumer electronic products like toasters and refrigerators
- One of the first project developed using java
  - A personal handheld remote control names start 7.
- At the same time world wide web and internet was gaining popularity. Gosling *et. Al.* realized that Java could be used for Internet programming.

Phases of Java Program

- The following figure describes the process of compiling and executing a java program.



Phases of Java Program (Contd)

The first step in creating a Java program is by writing your programs in a text editor. Examples of text editors you can use are notepad, vi, emacs, etc. This file is stored in a disk file with the extension `.java`.

After creating and saving your Java program, compile the program by using the Java Compiler. The output of this process is a file of Java **bytecodes** with the file extension `.class`.

The `.class` file is then interpreted by the Java interpreter that converts the bytecodes into the machine language of the particular computer you are using.

Task	Tool to Use	Output
Write the program	Any text Editor	File with .java extension
Compile the program	Java Compiler ***javac Myprogram.java***	File with .class extension (Java bytecodes)
Run the program	Java Interpretor ***java MyProgram***	Program output

Hello World Example

```
// This is an example of a single line comment using two slashes

/* This is an example of a multiple line comment using the slash and asterisk.
   This type of comment can be used to hold a lot of information or deactivate
   code but it is very important to remember to close the comment. */

/**
 * This is an example of a Javadoc comment; Javadoc can compile documentation
 * from this text.
 */

class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!"); // Display the string.
    }
}
```

Integrated Development Environments

- Note that taste in IDEs is a highly personal matter, but Eclipse, JBuilder, and Sun Java Studio (in that order) appear to be the most popular choices.
- **Eclipse**

  The Eclipse IDE for Java Developers contains what you need to build Java applications. The Eclipse IDE for Java Developers provides superior Java editing with validation, incremental compilation and much more.
- For downloading Eclipse,please visit http://www.eclipse.org/downloads/ and download Eclipse IDE for Java Developers
- **JCreator**

  JCreator is a powerful IDE for Java. JCreator is the development tool for every programmer that likes to do what he does best: programming. It is faster, more efficient and more reliable than other Java IDE's. Therefore it is the perfect tool for programmers of every level, from learning programmer to Java-specialist.

THANK YOU