School of Science & Engineering
Department of CSE
Canadian University of Bangladesh

Lecture-1: Object-Oriented Programming (OOP)

Prerequisite: CSE 1101

Semester: Summer 2024

Instructor: Prof. Miftahur Rahman, Ph.D.

Email: miftahur.rahman@cub.edu.bd

Office hours:

- Sun, Mon: 10:00 - 11:30
- Wed: 10:00 – 11:30
- By appointment.

Lectures hours:

Tues:1:10 pm-2:40 pm ;

Thrs:1:10 pm-2:40 pm;

Thrs:2:40 pm-4:00 pm

Prerequisites

Students are expected to have introductory programming experience, such as might be acquired from CSE 1101. Students should be able to use a terminal or command line program to perform the following types of actions:

- View contents of a directory/folder (e.g., use dir, ls)
- Navigate to a directory/folder (e.g., use cd)
- Create a new director/folder (e.g., use mkdir)

Grading Policy

- Attendance: **10%**
- Coding Assignments: **10%**
- Quizzes: **10%**
- Midterm Exam: **30%**
- Final Exam: **40%**

Rationale of OOP in Java:

The "OOP in Java" course is designed to provide students with essential programming skills, foundational knowledge, and practical experience necessary for a successful career in software development. By focusing on industry-relevant skills, problem-solving abilities, and best practices, the course ensures that students are well-equipped to meet the demands of the ever-evolving technology landscape.

Text & Reference Books:

- Text Book : Y. Daniel Liang, "Introduction to Java Programming", 10th edition, Prentice Hall.

- Reference Book: Paul Deitel & Harvey Deitel " Java How to Program", 10th edition, Pearson.

Course Outline

- Introduction
- Programming Fundamentals
- Methods
- Arrays
- Objects and Classes
- Inheritance and Polymorphism
- Abstract Classes and Interfaces
- Generics
- Exception Handling and Text I/O
- Introduction to JavaFX (GUI)

Academic Honesty

- You are allowed to discuss assignments with other students in the class.
  - Getting verbal advice/help from people who have already taken the course is also fine.
- However, any sharing of code is <u>not acceptable</u>.
  - Under no circumstances may you hand in work done with or by someone else under your own name.

Learning Outcomes

After finishing the course you will learn four important pillars of OOP (encapsulation, inheritance, abstraction, and polymorphism)

To understand the four pillars in Java, you will:

- Learn the history of Java
- Learn the basic elements of a Java program
- Learn how to execute Java programs
- Understand the motivation behind object-oriented programming
- Understand how to think of solutions in terms of classes and objects
- Understand how whitespace, commenting, errors, variables, types, expressions, and casting manifest in Java
- Learn how to create objects of existing classes
- Learn how to invoke methods of existing classes
- Create objects and invoke methods of the String class
- Learn how to take inputs from the terminal
- Learn how to specify formatting requirements for text printed to the terminal
- Learn three kinds of decision-making statements (if, if-else, switch)
- Learn three kinds of iteration statements (while, do-while, for)
- Learn how to instantiate one- and two-dimensional arrays
- Learn how to access, change, traverse, and search for data in arrays
- Explore modularity and reusability in the context of methods
- Learn how to define static methods
- Understand when and how to overload methods

Java's Origin Story

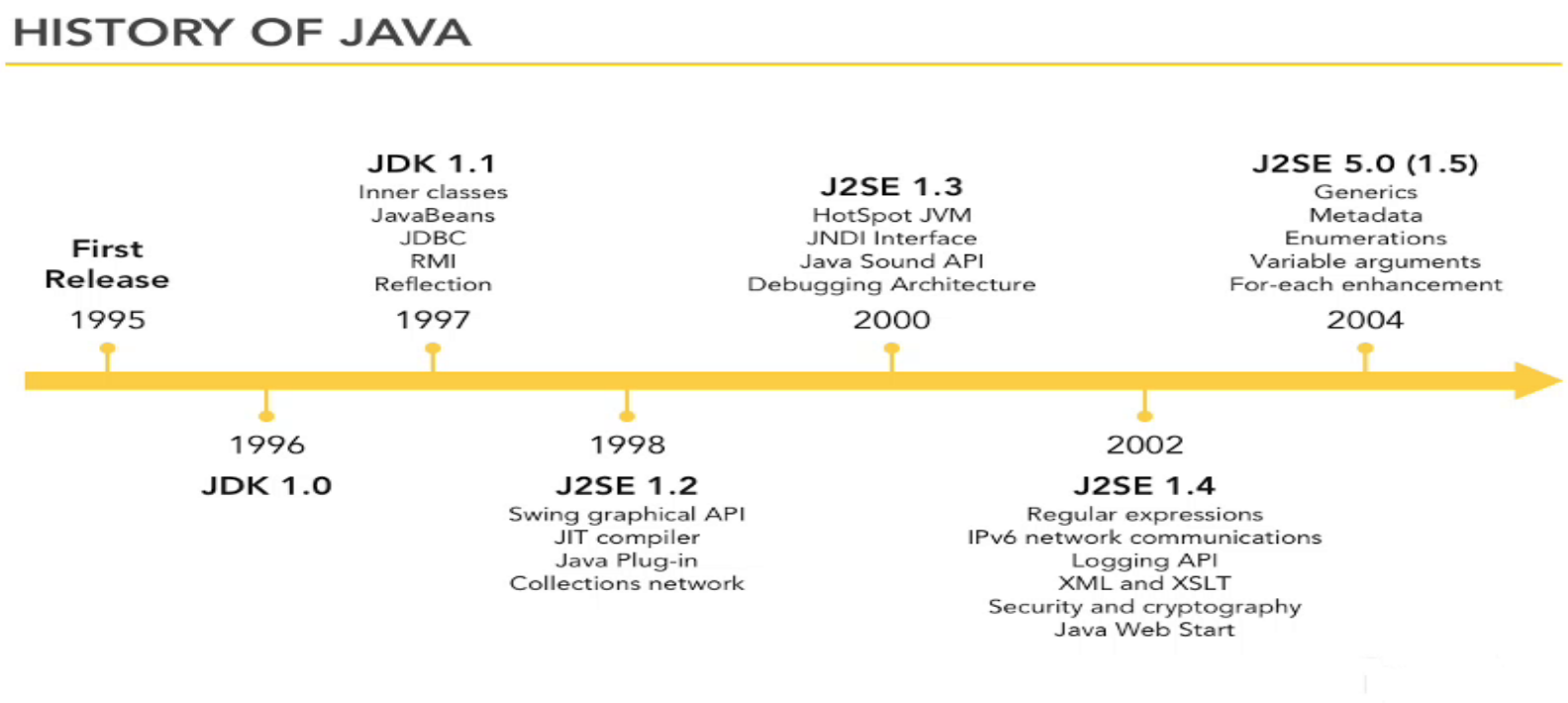
Java 1.0 was officially released in 1996 by a company called Sun Microsystems. It originated out of a need for a language to help write programs that run on appliances and other kinds of electronic devices, which were increasingly getting smarter as computing power became cheaper, faster, and smaller. Popular languages like C and C++ were already around for many years before Java, and C++ is even Object-Oriented. However, a group of scientists at Sun, who had been experimenting in this space, determined that existing languages needed to be safer in order to be deployed on devices, which sometimes controlled critical aspects of daily life.

While C and C++ are very efficient and flexible languages, it is quite possible for even experienced programmers to miss significant memory errors and security bugs that could cause a device to behave inconsistently, or even worse, fail during operation. Imagine an elevator regularly getting stuck because it could not fully execute the code that navigates it through floors due to a memory leak that its programmer did not notice. Or, if you don't mind tight spaces, imagine a microwave burning your dinner from your favorite takeout spot (and maybe more) because a memory issue caused it to overheat.

Although Java does not guarantee flawless programs, its programmers inherently avoid much of the memory and security issues that are common in languages like C and C++. The reason is primarily due to very powerful mechanisms, like automatic memory management, that its creators built into the language. Such features are therefore inherent to every Java program, regardless of the experience level of the programmer.

Fortunately, you don't have to be a device programmer in order to reap such benefits. During Java's development, its creators explored several different directions with the language, which greatly increased its scope and popularity. For example, they noticed the static nature of websites at the time (the 1990s) and developed tools for using Java to write programs that could actually be embedded on web-pages. Such programs, called applets, helped Java's adoption by allowing a large population of web programmers to create pages with dynamic content like games, stock tickers, animations, and much more.

Over time, Java evolved into a general-purpose programming language allowing programmers to write software applications that run on conventional computers like desktops and laptops. As illustrated in the graphic below, new features have been added to the language since version 1.0, and more are expected due to the significant amount of use and investment that has been put into it.



JDK: Java Development Kit,
JDBC: The Java Database Connectivity (JDBC) API provides universal data access from the Java programming language.
RMI: RMI stands for Remote Method Invocation. It is a mechanism that allows an object residing in one system (JVM) to access/invoke an object running on another JVM. RMI is used to build distributed applications; it provides remote communication between Java programs. It is provided in the package java.
Reflection: Reflection is a feature in the Java programming language. It allows an executing Java program to examine or "introspect" upon itself, and manipulate internal properties of the program. For example, it's possible for a Java class to obtain the names of all its members and display them.
Swing Graphical API: Swing API is a set of extensible GUI Components to ease the developer's life to create JAVA based Front End/GUI Applications.
JIT Compiler: The Just-In-Time (JIT) compiler is a component of the runtime environment that improves the performance of Java™ applications by compiling bytecodes to native machine code at run time.
Java Plug-in: Java Plug-in extends the functionality of a web browser, allowing applets or Java Beans to be run under Sun's Java 2 runtime environment (JRE) rather than the Java runtime environment that comes with the web browser. Java Plug-in is part of Sun's JRE and is installed with it when the JRE is installed on a computer. It works with both Netscape and Internet Explorer.

HotSpot JVM: The open source HotSpot JVM enables a Java program to run on different hardware, because JVMs exist for a variety of platforms.

JNDI Interface: The Java Naming and Directory Interface™ (JNDI) is an application programming interface (API) that provides naming and directory functionality to applications written using the Java™ programming language. It is defined to be independent of any specific directory service implementation. Thus a variety of directories--new, emerging, and already deployed--can be accessed in a common way.

JAVA Sound API: JavaSound is a collection of classes and interfaces for effecting and controlling sound media in java. It consists of two packages.

- ***javax.sound.sampled:*** This package provides an interface for the capture, mixing digital audio.
- ***javax.sound.midi:*** This package provides an interface for MIDI (Musical Instrument Digital Interface) synthesis, sequencing, and event transport.

IPV6 network communications: Internet Protocol version 4 (IPv4) has long been the industry standard version of the Internet Protocol (IP) for delivering data over the Internet. Internet Protocol version 6 (IPv6) is the next generation Internet layer protocol. Java applications support both IPv4 and IPv6 automatically.

XML , XSLT: XSLT is the XML Stylesheet Language Translator.

Today, Java is a product of Oracle Corporation, which acquired Sun Microsystems in 2010. Oracle claims that a few billion devices currently run Java--spanning servers, cell phones, ATMs, cable boxes, TVs and much more. It's likely that you've interacted with a Java-powered device, perhaps even without knowing.

In this course, we'll only focus on writing standalone applications for conventional computers. Applets, which were mentioned earlier, have been deprecated in the recent versions of the language.

Outline

- What is Programming (1.1)
- Programming Languages (1.3)
- Operating Systems (1.4)
- Procedural vs. object oriented programming.
- What is Java? (1.5)
- The Java Language Specification, API, JDK, and IDE (1.6)
- A simple Java program (1.7)
- Creating, Compiling, and Executing a Java Program. (1.8)
- Programming Errors (1.10)

What is Programming?

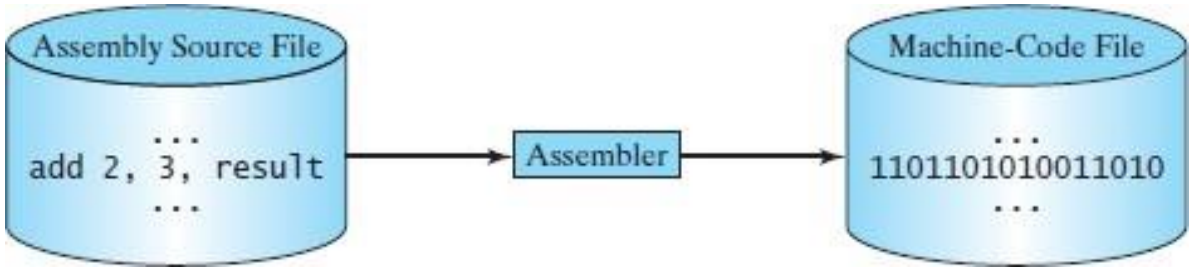
- Programming means to create software.
  - Software contains the instructions that tell a computer what to do.
- Software developers create software with the help of powerful tools called *programming languages*.

Evolution of Programming Languages Machine Language

- Computer's native language.
- Differs among different types of computers.
- Set of built-in primitive instructions in the form of binary code.
- For example to add two numbers, you might have to write something like this:
  - 1110110001100110
- Difficult to read and modify.

Evolution of Programming Languages
Assembly Language

- Were created as an alternative to machine languages.
- Uses short descriptive words.
- One-to-one correspondence between each assembly instruction and machine instruction.
- An *assembler* is used to translate assembly-language programs into machine code.

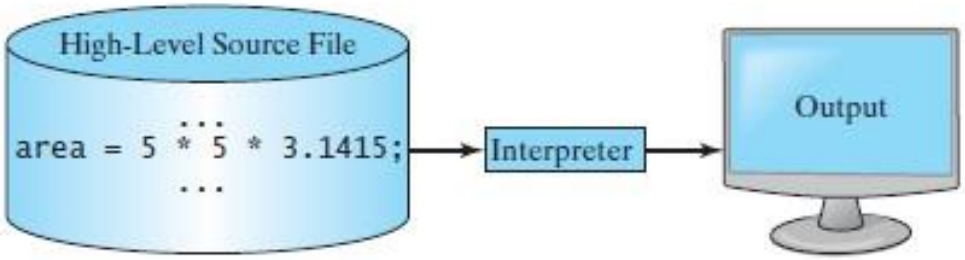


Evolution of Programming Languages High-Level Language

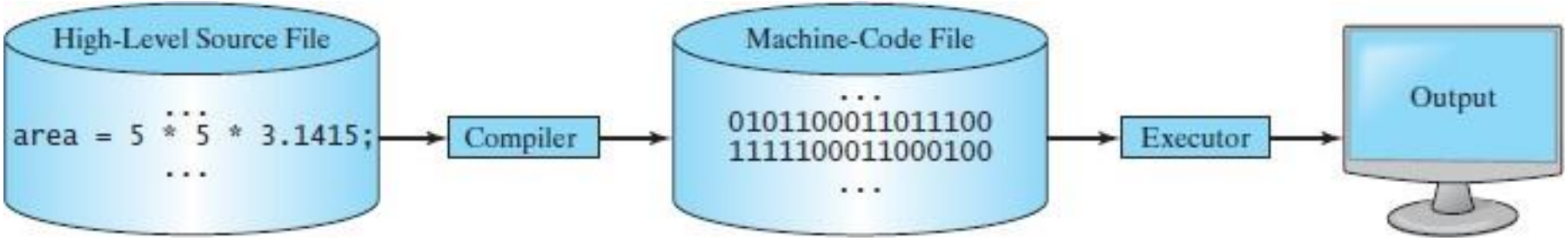
- Platform independent.
  - You can write a program in a high-level language and run it on different types of machines.
- Easy to learn and use.
- Examples of high level languages:
  - BASIC, C/C++, C#, COBOL, FORTRAN, Java, Python, R.
- An *interpreter* or a *compiler* is used to translate the *source code* into machine code.
  - *What is the difference between interpreters and compilers?*

Evolution of Programming Languages High-Level Language (Cont.)

- Interpreters read one statement from the source code, translate it to machine code and execute it right away.



- Compilers translate the entire source code into a machine code, and the machine file is then executed.

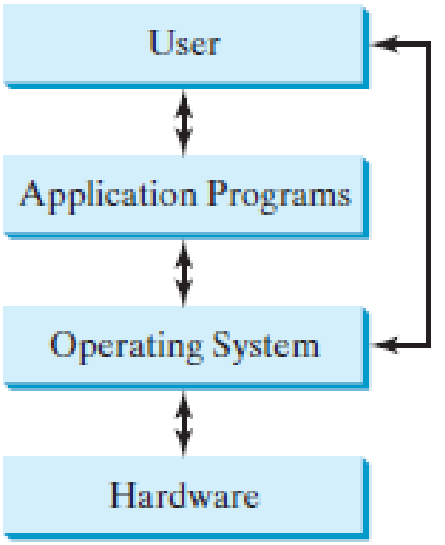


Java Editions

- Java *Standard Edition* (Java SE) to develop client- side standalone applications or applets.
  - The foundation upon which all other Java technology is based.
  - There are many versions of Java SE. The latest is Java SE 21.
- Java *Enterprise Edition* (Java EE) to develop server-side applications.
- Java *Micro Edition* (Java ME) to develop applications for mobile devices.

Operating Systems

- The operating system is the most important program that runs on the computer.
- Popular operating systems for general-purpose computers include:
  - Microsoft Windows, Mac OS, and Linux.
- Major tasks of an operating system:
  - Controlling and monitoring system activities.
  - Allocating and assigning system resources.
  - Scheduling operations.



Procedural vs. Object Oriented Programming

- Procedural programming consists of designing a set of modules (*functions* or *methods*) to solve a problem.
  - Focuses on how to manipulate the data, then on what data structures to use to make the manipulation easier.
- Object oriented programming puts data first, then look at algorithms that operate on the data.
  - Focuses on objects and operations on objects.

What is Java?

- Was developed by a team led by *James Gosling* at Sun Microsystems.
- Powerful and versatile programming language for developing software running on:
  - Mobile devices.
  - Desktop computers.
  - Servers.

The Java Language Specification

- Programming languages have strict rules of usage.
- The *Java Language Specification* is a technical definition of the Java programming language syntax and semantic.
  - https://docs.oracle.com/javase/specs/

Java API, JDK, and IDE

- The *Application Program Interface* (*API*):
  - Also known as library.
  - Contains predefined classes and interfaces for developing Java programs.
- The *Java Development Kit* (*JDK*) consists of a set of separate programs, each invoked from command line, for developing and testing Java programs.
- The *Integrated Development Environment* (*IDE*) such as (NetBeans, Intelli J IDEA, BlueJ, Eclipse, DrJava, J Developer, Jcreator, jGRASP, MyEclipse, Visual Studio, Greenfoot, Androidstudio, Xcode, AppCode, Codenvy, PyCharm, etc.) provides