CIS016-1 – Principles of Programming / CIS096-1 – Principles of Programming and Data Structures / PAT001-1 – Principles of Programming

Week 1 – Introduction to Programming (Programming in Java)

*

*

University of Bedfordshire

Agenda

- Introduction to Programming
- Programming languages
- Why Java?
- Writing Java programs

*

University of Bedfordshire

What Is A Computer?

- A computer is
  - an electronic device,
  - operating under the control of instructions (**software**) stored in its own memory unit,
  - that can accept data (**input**), manipulate data (**process**), and produce information (**output**) from the processing.
- Generally, the term is used to describe a **collection of devices that function together as a system**.

Devices that comprise a computer system

*

What Does A Computer Do?

- Computers can perform **four general operations**, which comprise the **information processing cycle**:
  - Input
  - Process
  - Output
  - Storage

University of Bedfordshire

Data and Information

- All computer processing requires **data**, which is
  - a collection of raw facts, figures and symbols, such as numbers, words, images, video and sound,
  - given to the computer during the input phase

- Computers manipulate data to create information. **Information** is data that is organized, meaningful, and useful.

*

How do Computers know what to do?

- It must be given a detailed list of instructions, called a **computer program** or **software**, that tells it exactly what to do
- Before processing a specific job, the computer program corresponding to that job must be stored in **memory**
- Once the program is stored in memory the computer can start the operation by
  - **executing the program instructions**
  - one after the other.

*

University of Bedfordshire

Computer Program (Java)

```
import java.io.*;

public class HelloWorld {
   public static final void main(String[] str) {
      System.out.println("Hello World!");
   }
}
```

Computer Program (Basic, early 80s)

```
10   PRINT "Hello World!"
 20 GOTO 10
```

Computer Program (Assembler, 6502, Apple II)

```
     .or $300
main     ldy #$00
.1       lda str,y
         beq .2
         jsr $fded ; ROM routine, COUT, y is preserved
         iny
         bne .1
.2       rts
str      .as "HELLO WORLD"
         .hs 0D00
```

See also
https://en.wikibooks.org/wiki/Computer_Programming/Hello_world

University of Bedfordshire

A Layered View of the Computer

Application Programs

Word-Processors, Spreadsheets,
Database Software, IDEs,
etc…

System Software

Compilers, Interpreters, Preprocessors, etc.
Operating System, Device Drivers

Machine with all its hardware

University of Bedfordshire

Operating System (OS)

- Provides several essential services:
  - Loading and running application programs
  - Allocating memory and processor time
  - Providing input and output facilities
  - Managing files of information

University of Bedfordshire

Application Programs

- Programs are written in **programming languages (PL)**
  - Pieces of the same program can be written in different PLs
    - Languages closer to the machine can be more efficient
    - As long as they agree on how to communicate

- A PL is
  - A special purpose and limited language
  - A set of rules and symbols used to construct a computer program
  - A language used to interact with the computer

University of Bedfordshire

Computer Languages

- **Machine Language**
  - Uses binary code, machine-dependent, not portable
- **Assembly Language**
  - Low-level Language. Uses mnemonics, machine-dependent, not usually portable
- **High-Level Language (HLL)**
  - Uses English-like language, machine independent,
  - Portable (but must be compiled for different platforms)
  - Examples: Pascal, C, C++, Java, Fortran, . . .

University of Bedfordshire

Types of Programming

- **Logic Programming:**
  - Operate on facts and relationships from which they can draw a coherent and simple conclusion
  - Examples: Prolog, Datalog
- **Functional Programming:**
  - More closely related to the mathematical concept of `function' than imperative programming languages
  - Examples: Lisp, Scheme, Erlang
- **Imperative Programming:**
  - Associated with languages like C, Fortran, Pascal etc.
- **Concurrent Programming:**
  - Concurrent programming is characterized by programming with more than one process
- **Object-Oriented Programming:**
  - The method of implementing programs which are organized as cooperative collections of objects.

University of Bedfordshire

Imperative Programming Languages

FORTRAN - old but still popular with scientists and engineers, new versions are introduced every few years.

COBOL - a still widely used and well-standardised language used in commerce.

Algol - an elegant, little-used and internationally designed language whose features were incorporated into other languages.

BASIC - often used by beginners on home computers.

University of Bedfordshire

Imperative Programming Languages

APL - an interactive scientific language with a very mathematical notation.

PL/1 - a failed attempt by IBM at achieving an all-purpose language, now almost dead.

Pascal - good for beginners, often taught as a first language.

Modula2 - a development of Pascal to make it more realistic for large programs, and to enable modern program design techniques to be used.

University of Bedfordshire

Imperative Programming Languages

C - a practical, systems language.

C++ - an object oriented development of C.

Java - a robust, secure, portable, network enabled variation of C/C++ (also **object-oriented**)

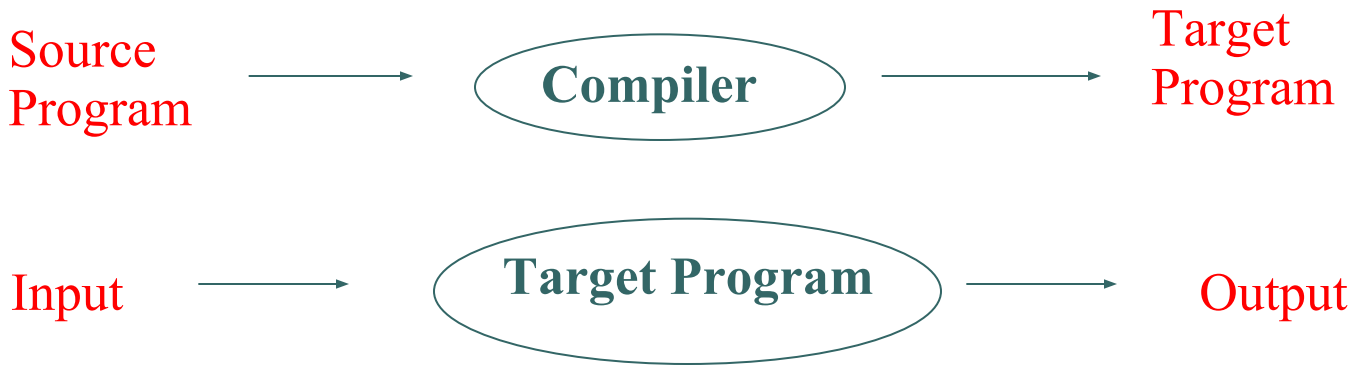
University of Bedfordshire

Ways of Executing Programs

- Basically there are two ways of executing a program:
  - Compilation
  - Interpretation
- Mixed forms are possible (ie. Java)

University of Bedfordshire

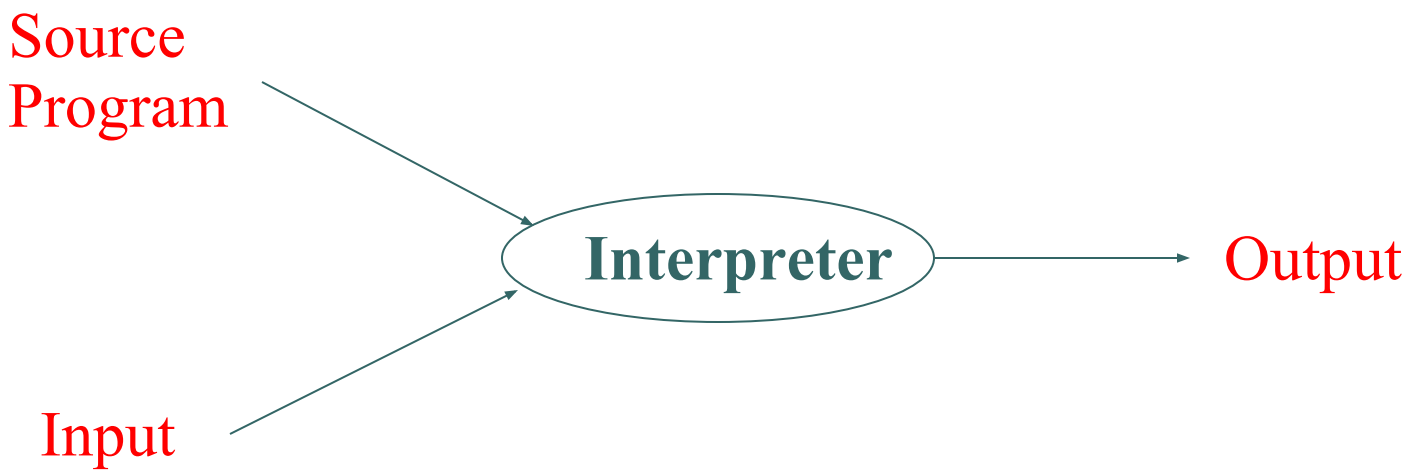
Compilation



- **Compiler** translates **source** (a Java, C, … program) into **target** (a machine language program)
- Compiler is itself a machine language program, presumably created by compiling some other high-level program
- Machine language, when written in a format understood by the OS is **object code**
- The compiled **target program** is executed; it takes some **input** and produces some **output**

University of Bedfordshire

Interpretation



- The **interpreter** stays around during execution
- It **reads and executes** statements **one at a time**

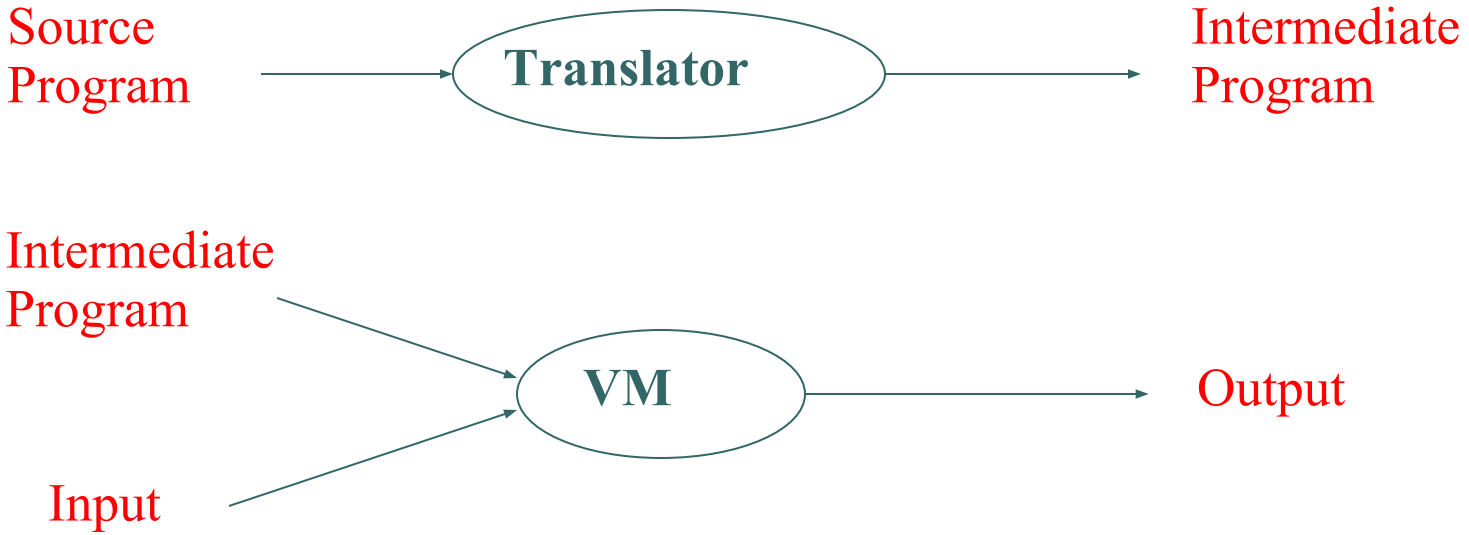
University of Bedfordshire

Compilation vs. Interpretation

- **Compilation**:
  - Syntax errors caught before running the program
  - Better performance
  - Decisions made once, at compile time
- **Interpretation**:
  - Better diagnostics (error messages)
  - More flexibility
  - Supports **late binding** (delaying decisions about program implementation until runtime)
    - Can better cope with PLs where type and size of variables depend on input
  - Supports creation/modification of program code on the fly (ie. Lisp, Prolog)

University of Bedfordshire

Compiler and Interpreter

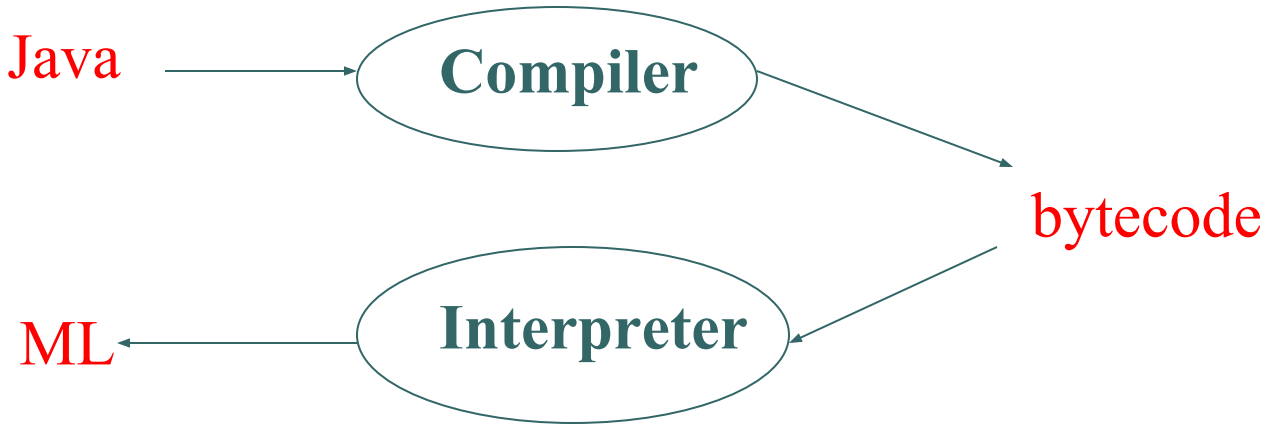


- Many programming languages implement this
- Interpreter implements a **Virtual Machine (VM)**.

University of Bedfordshire

Java

For portability:



ML:= Machine Language

University of Bedfordshire

Why Java?

- Java
  - General purpose language with good Internet features
  - Widely used
  - Virtual Machines available for major OSes
  - Simple enough as first language and OOP
  - Open source

University of Bedfordshire

Characteristics of Java

- Java is simple
- Java is object-oriented
- Java is distributed
- Java is interpreted
- Java is robust
- Java is secure
- Java is architecture-neutral
- Java is portable
- Java's performance
- Java is multithreaded
- Java is dynamic

University of Bedfordshire

History

- James Gosling and Sun Microsystems
- Java, May 20, 1995, Sun World
- HotJava
  - The first Java-enabled Web browser

*

University of Bedfordshire

Java Editions

- Java Standard Edition (J2SE)
  - J2SE can be used to develop **client-side standalone applications** or **applets**
- Java Enterprise Edition (J2EE)
  - J2EE can be used to develop **server-side applications** such as Java servlets and Java ServerPages
- Java Micro Edition (J2ME)
  - J2ME can be used to develop applications for **mobile devices** such as cell phones.
- Java Android SDK
  - Most **Android** apps are written in Java

*

University of Bedfordshire

Java Version History (J2SE)

- JDK 1.0 (1996)
- JDK 1.1 (1997)
- J2SE 1.2 (a.k.a JDK 1.2, 1998)
- J2SE 1.3 (a.k.a JDK 1.3, 2000)
- J2SE 1.4 (a.k.a JDK 1.4, 2002)
- J2SE 5.0 (a.k.a. JDK 1.5, 2004)
- Java SE 6 (a.k.a. JDK 1.6.0, 2006)
- Java SE 7 (a.k.a. JDK 1.7, 2011)
- Java SE 8 (2014)
- ...
- Java SE 18 (March, 2022)

University of Bedfordshire

To write your own Java program

- Take a simple text editor (not Word!) and a console (Windows command prompt, Unix shell)

- Use of an **Integrated Development Environment (IDE)**
  - IDEs help programmers – context sensitive, project control
  - Examples
    - **Visual Studio Code**
    - **BlueJ**
    - **EditPlus**
    - **Eclipse**

*

Hello World!

`System` comes from the java.io package

Name of program/class

The `main` method

```
import java.io.*;

public class HelloWorld {
    public static final void main(String[] str) {
        System.out.println("Hello World!");
    }
}
```

- Save file to `HelloWorld.java` (class name and file name must be identical!)
- Compile file with `javac HelloWorld.java`
- Execute file with `java HelloWorld`

University of Bedfordshire

Hands On Java

- Try Java yourself - there's nothing like practice when learning a new programming language!
- Just play around, don't be put off when something doesn't work – it happens to all of us!
- And it's more fun, too! ☺

University of Bedfordshire

Task 1

- HelloWorld and BlueJ
  - Look at the supplementary material on BREO (under ‘Guided Learning’). You find a YouTube video there. Watch it.
  - Start a new project called HelloWorld or something similar, using BlueJ.
  - Create a new class for the HelloWorld entity
  - Copy the code in the HelloWorld.java file and amend with your name etc.
  - Compile and run.

University of Bedfordshire

Task 2 (not only for today)

- Java Language
  - Make notes concerning aspects of Java as indicated during the lecture

University of Bedfordshire

Task 3

- Directed learning
  - In the 12 hours that you should be attempting outside of these scheduled sessions please read Chapter 1 of **Absolute Java** by **Savitch**