Lecture 1
Introduction to Java

MIT- AITI 2004
What is a Computer Program?

• For a computer to be able to do anything (multiply, play a song, run a word processor), it must be given the instructions to do so.

• A program is a set of instructions written by humans for computers to perform tasks.

• The instructions are written in programming languages such as C, C++, Java, etc.
**Recipe Analogy**
Comparing a computer program to a food recipe

<table>
<thead>
<tr>
<th>Food Recipe</th>
<th>Computer Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>• a <em>chef</em> writes a set of instructions called a <em>recipe</em></td>
<td>• a <em>programmer</em> writes a set of instructions called a <em>program</em></td>
</tr>
<tr>
<td>• the recipe requires specific <em>ingredients</em></td>
<td>• the program requires specific <em>inputs</em></td>
</tr>
<tr>
<td>• the <em>cook</em> follows the instruction step-by-step</td>
<td>• the <em>computer</em> follows the instructions step-by-step</td>
</tr>
<tr>
<td>• the <em>food</em> will vary depending on the <em>amount of ingredients</em> and the <em>cook</em></td>
<td>• the <em>output</em> will vary depending on the <em>values of the inputs</em> and the <em>computer</em></td>
</tr>
</tbody>
</table>
Compiling Programs

• Computers do not understand the languages (C++, Java, etc) that programs are written in.

• Programs must first be compiled (converted) into machine code that the computer can run.

• A compiler is a program that translates a programming language into machine code.
Running Programs

• All programs follow a simple format:
  Input → Execution → Output

• Inputs can be from users, files, or other computer programs

• Outputs can take on many forms: numbers, text, graphics, sound, or commands to other programs
Multiple Compilers

- Because different operating systems (Windows, Macs, Unix) require different machine code, you must compile most programming languages separately for each platform.
Java Interpreter

- **Java** is a little different.
- Java compiler produces **bytecode** not machine code.
- Bytecode can be run on any computer with the **Java interpreter** installed.
Advantages of Java...

Advantages:

• Java is platform independent. Once it's compiled, you can run the bytecode on any machine with a Java interpreter. You do not have to recompile for each platform.

• Java is safe. Certain common programming bugs and dangerous operations are prevented by the language and compiler.

• Java standardizes many useful operations like managing network connections and providing graphical user interfaces.
... and disadvantages of Java

Disadvantages:

- Running bytecode through the interpreter is not as fast as running machine code, which is specific to that platform.

- Because it is platform independent, it is difficult to use platform specific features (e.g., Windows taskbar, quick launch) in Java.

- Java interpreter must be installed on the computer in order to run Java programs.
Your First Java Program

• Open Notepad and type the following piece of Java code exactly:

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

• Save this file as HelloWorld.java (watch capitalization) in the following directory:
  
c:\java
Compiling and Running Your First Program

• Open the command prompt in Windows
• To run the program that you just wrote, type at the command prompt:
  
  cd c:\java

• Your command prompt should now look like this:
  
  c:\java>

• To compile the program that you wrote, you need to run the Java Development Tool Kit Compiler as follows:
  At the command prompt type:
  
  c:\java> javac HelloWorld.java

• You have now created your first compiled Java program named HelloWorld.class
• To run your first program, type the following at the command prompt:
  
  c:\java>java HelloWorld

Although the file name includes the .class extension, this part of the name must be left off when running the program with the Java interpreter.
Congratulations!

You’ve created your first Java program!
Object-Oriented Programming

• Java is an **object-oriented** programming language

• For the rest of this lecture, we’ll introduce you to the basic principles of object-oriented programming.

• We won’t be using these principles immediately, but they will become important over the next few weeks.
OOP Concepts

• In object-oriented programming (OOP), programs are organized into objects

• The properties of objects are determined by their class

• Objects act on each other by passing messages
Object

• **Definition:** An object is a software bundle that has *State* and *Behavior*.

• Software Objects are often used to model real-world objects.

• Example: dogs have states (name, color, hungry, breed) and behaviors (bark, fetch, and wag tail).
Object Examples

• Example 1: Dogs
  – States: name, color, breed, and “is hungry?”
  – Behaviors: bark, run, and wag tail

• Example 2: Cars
  – States: color, model, speed, direction
  – Behaviors: accelerate, turn, change gears
Class

• **Definition**: A class is a blueprint that defines the states and the behaviors common to all objects of a certain kind.

• In the real world, you often have many objects of the same kind. For example, a guard dog, herding dog, snoop dog . . .

• Even though all dogs have four legs, and bark, each dog’s behavior is independent of other dogs.

• For example: Dog #1 is a black Poodle, Dog #2 is a red Irish Setter
Message

• **Definition**: Software objects interact and communicate with each other by sending *messages* to each other.

• **Example**: when you want your dog to gather a herd of goats, you whistle and send him out.
Summary of OOP

• When writing an object-oriented program, we define **classes**, which describe categories of objects, and the states and behaviors that they have in common.

• We then create **objects** which belong to classes, and share the common features of their class.

• Objects interact with each other by passing **messages**.

• You will be creating your own classes and objects soon!