Introduction to Java

Programs

Computer ***programs***, known as ***software***, are instructions to the computer.

You tell a computer what to do through **programs**.

Programs are written using programming languages.

Programming Languages

Machine Language   Assembly Language   High-Level Language

```
1101101010011010
```

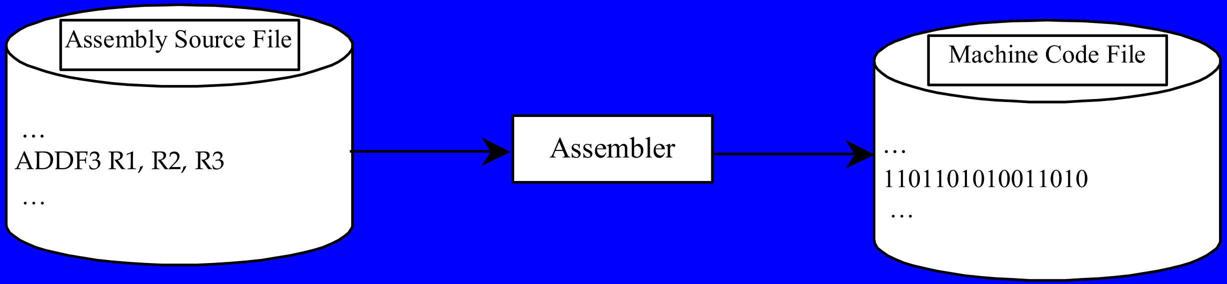
```
Computers can run instructions only
  in machine language!
```

Programming Languages

Machine Language   Assembly Language   High-Level Language

Humans are not comfortable with machine languages, so they made low-level languages like Assembly.

ADDF3 R1, R2, R3



Programming Languages

Machine Language   Assembly Language   High-Level Language

Then high-level languages were invented which are English-like and easy to learn and program.

```
Sum = 4 + 5;
Area = 5 * 5 * 3.1415;
```

Popular High-Level Languages

Language	Description
Ada	Named for Ada Lovelace, who worked on mechanical general-purpose computers. The Ada language was developed for the Department of Defense and is used mainly in defense projects.
BASIC	Beginner's All-purpose Symbolic Instruction Code. It was designed to be learned and used easily by beginners.
C	Developed at Bell Laboratories. C combines the power of an assembly language with the ease of use and portability of a high-level language.
C++	C++ is an object-oriented language, based on C.
C#	Pronounced "C Sharp." It is a hybrid of Java and C++ and was developed by Microsoft.
COBOL	COmmon Business Oriented Language. Used for business applications.
FORTRAN	FORmula TRANslation. Popular for scientific and mathematical applications.
Java	Developed by Sun Microsystems, now part of Oracle. It is widely used for developing platform-independent Internet applications.
Pascal	Named for Blaise Pascal, who pioneered calculating machines in the seventeenth century. It is a simple, structured, general-purpose language primarily for teaching programming.
Python	A simple general-purpose scripting language good for writing short programs.
Visual Basic	Visual Basic was developed by Microsoft and it enables the programmers to rapidly develop graphical user interfaces.

Interpreting/Compiling Source Code

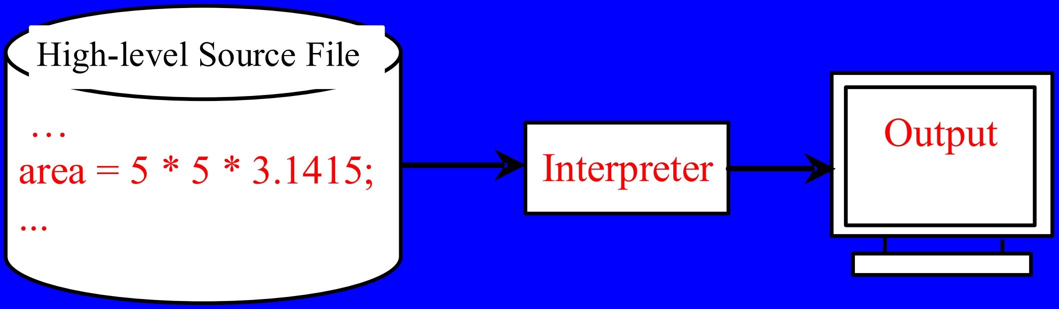
A program written in a high-level language is called a ***source program*** or ***source code***.

Source Code → Machine Code (Or Virtual Machine Code) → Execute

The translation can be done using another programming tool called an ***interpreter*** or a ***compiler***.

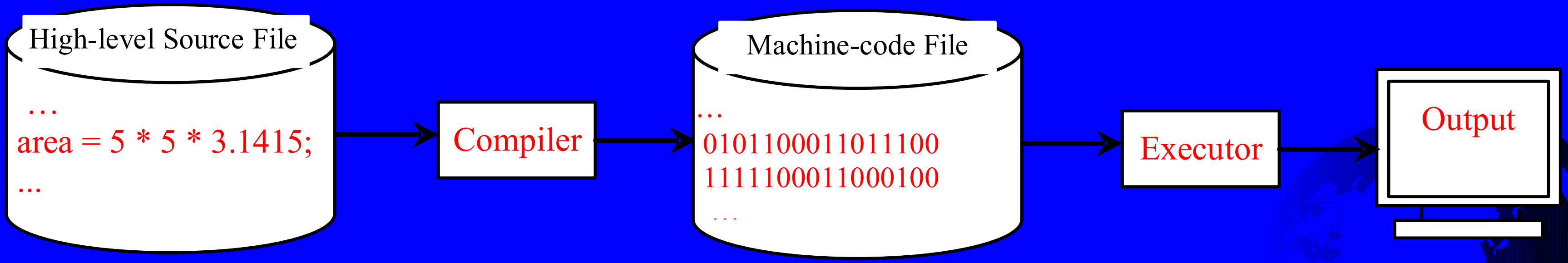
Interpreting Source Code

An interpreter reads one statement from the source code, translates it to the machine code or virtual machine code, and then executes it right away.



Compiling Source Code

A compiler translates the entire source code into a machine-code file, and the machine-code file is then executed, as shown in the following figure.



A Simple Java Program

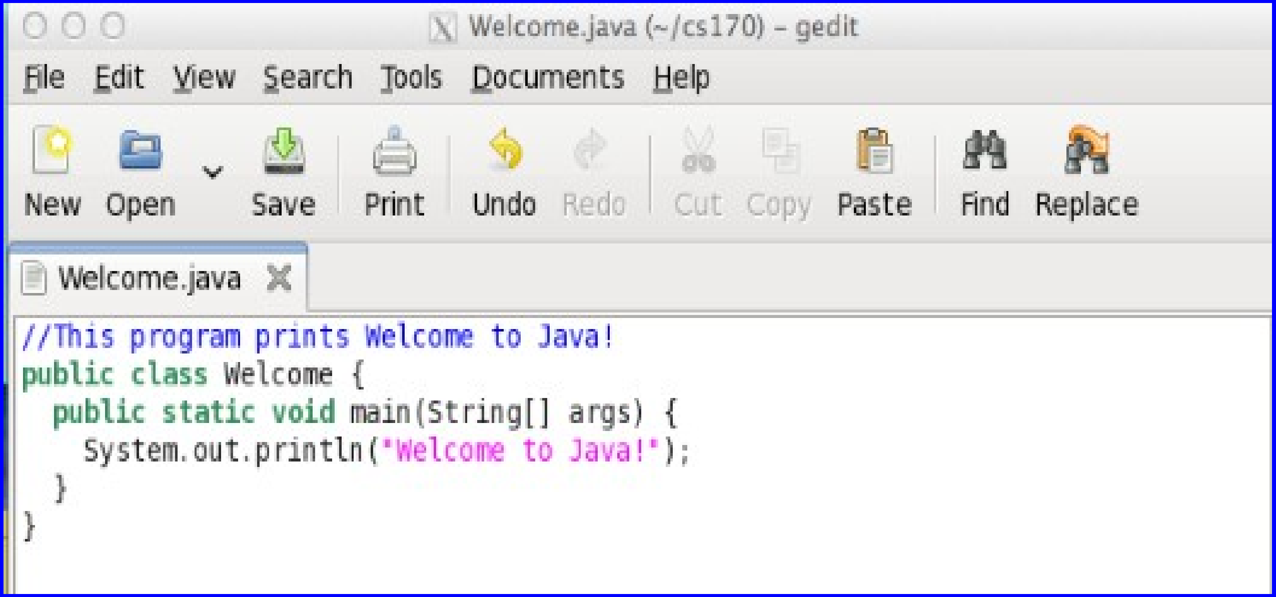
```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Creating and Editing Using gedit

To use , type

```
gedit  Welcome.java
```

from the terminal.


Welcome.java (~/cs170) – gedit
File Edit View Search Tools Documents Help
New Open Save Print Undo Redo Cut Copy Paste Find Replace
Welcome.java

```java
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Compiling Java program

```
javac  Welcome.java
```

javac is the java compiler
It translates Java source code to java bytecode
(a type of virtual machine code)

Running Java program

```
java  Welcome
```

Java Virtual Machine is a software that can execute Java bytecode.

Trace a Program Execution

Enter main method

```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

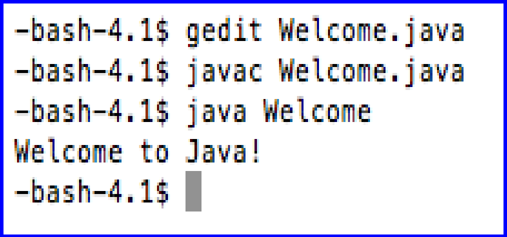
Trace a Program Execution

Execute statement

```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Trace a Program Execution

```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```



```
-bash-4.1$ gedit Welcome.java
-bash-4.1$ javac Welcome.java
-bash-4.1$ java Welcome
Welcome to Java!
-bash-4.1$
```

print a message to the console

Hierarchy of a Java Program

A book (program) consists of a number of chapters (classes)

Each chapter (class) consists of a number of paragraphs (methods)

Each paragraph (method) consists of a number of sentences (statements)

Each sentence (statement) must obey the syntax rules in the English (Java) language

Class

Every Java program must have at least one class. Each class has a name. By convention, class names start with an uppercase letter. In this example, the class name is Welcome.

```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Method

A method contains a set of statements!

In order to run a class, the class must contain a method named main. The program is executed from the main method.

```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Statement

A statement represents an action or a sequence of actions.

```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Statement Terminator

Every statement in Java ends with a semicolon (;).

```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

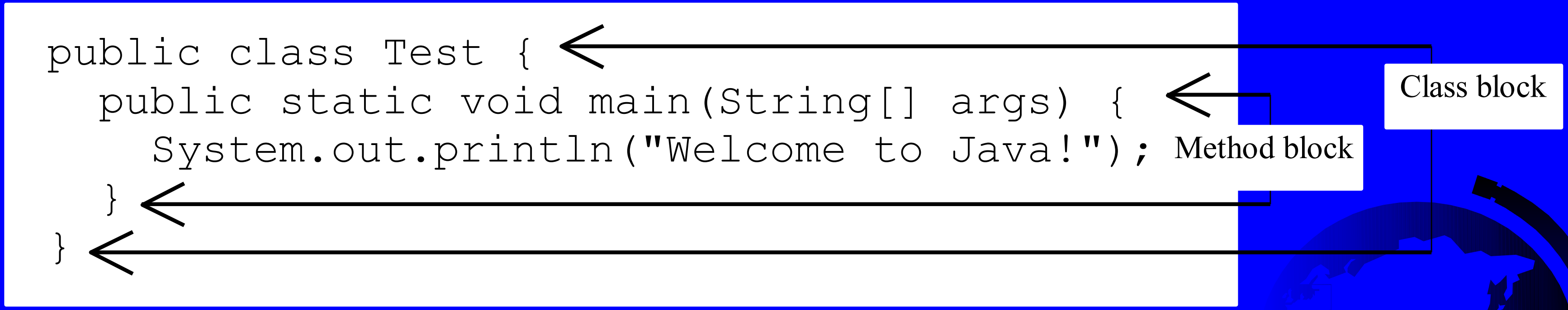
Reserved words

Reserved words or keywords are words that have a specific meaning to the compiler and cannot be used for other purposes in the program.

```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Blocks

A pair of braces in a program forms a block that groups components of a program.



```
public class Test {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Special Symbols

Character	Name	Description
{}	Opening and closing braces	Denotes a block to enclose statements.
()	Opening and closing parentheses	Used with methods.
[]	Opening and closing brackets	Denotes an array.
//	Double slashes	Precedes a comment line.
" "	Opening and closing quotation marks	Enclosing a string (i.e., sequence of characters).
;	Semicolon	Marks the end of a statement.

{ ... }

```
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

(...)

```
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

;

```
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

// ...

```
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

" ... "

```
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Programming Style and Documentation

Appropriate Comments

Naming Conventions

Proper Indentation and Spacing Lines

Block Styles

Appropriate Comments

Include a summary at the beginning of the program to explain what the program does, its key features, its supporting data structures, and any unique techniques it uses.

Naming Conventions

Choose meaningful and descriptive names.

Class names:

- Capitalize the first letter of each word in the name. For example, the class name `ComputeExpression`.

Proper Indentation and Spacing

Indentation

– Indent two spaces.

Spacing

– Use blank line to separate segments of the code.

Block Styles

Use end-of-line style for braces.

Next-line style

```
public class Test
{
  public static void main(String[] args)
  {
    System.out.println("Block Styles");
  }
}
```

End-of-line style

```
public class Test {
  public static void main(String[] args) {
    System.out.println("Block Styles");
  }
}
```

JDK (Java Development Kit)

- The Java Development Kit (JDK) is a software development environment used for developing Java applications.

- It includes the Java Runtime Environment (JRE), an executer/launcher (java), a compiler (javac), an archiver (jar), a documentation generator (javadoc) and other tools needed in Java development.

- Newest Version:

  JDK 1.7 (2011) a. k. a. JDK 7 or Java 7

JDK 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, Java ServerPages, and Java ServerFaces.
- Java Micro Edition (J2ME).
  - J2ME can be used to develop applications for mobile devices such as cell phones.