Computer Programming

- A (high level) *programming language* is a language used by people to tell a computer what to do.
- These languages are used to write *source code*, which consists of a series of *statements*: small tasks for the computer to perform, such as drawing a rectangle or adding two numbers.
- A *compiler* or *interpreter* is used to convert the human-written source code into low level *machine code* the computer can use.

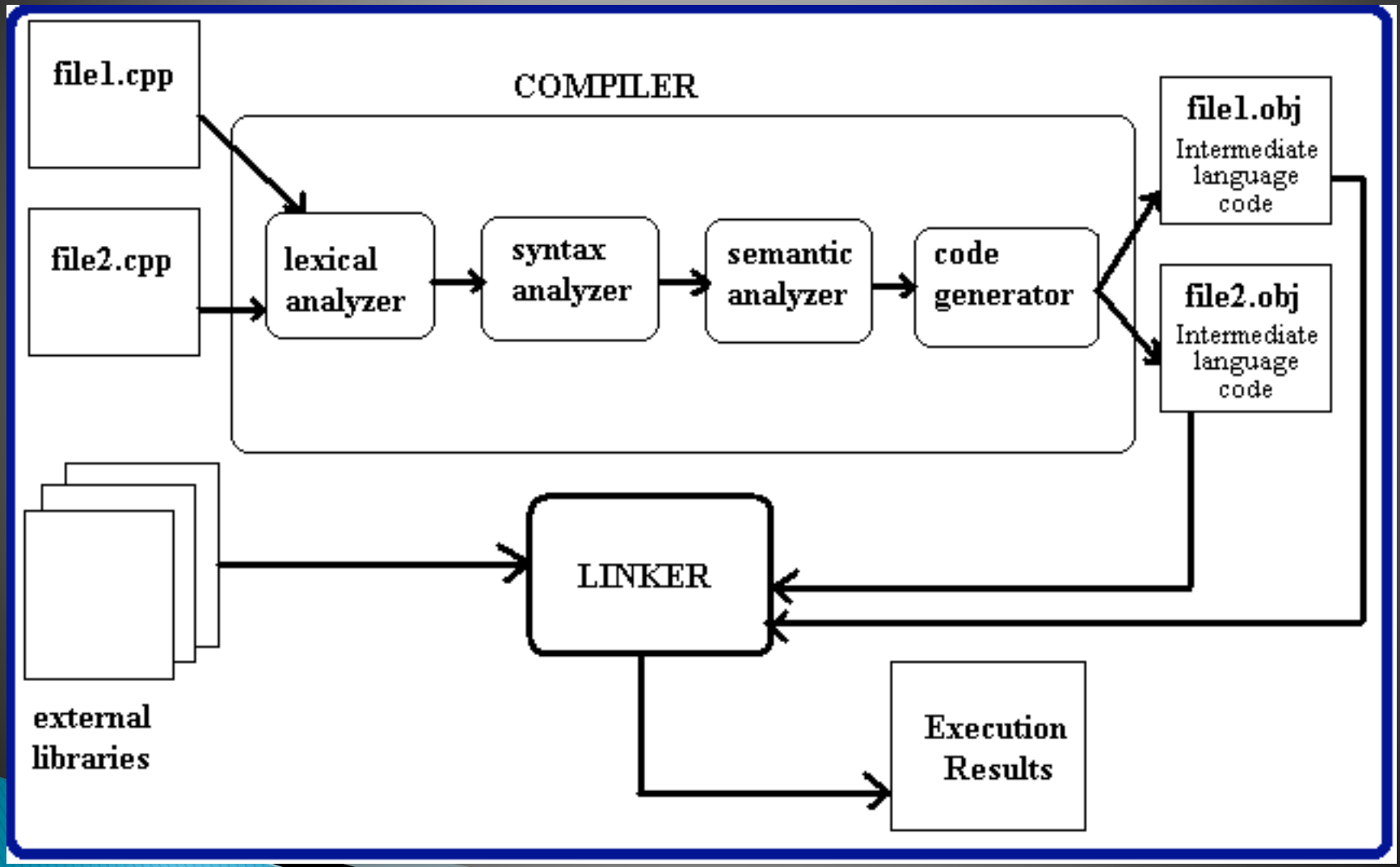
Computer Programming

- Machine code is written in binary, a series of 0's and 1's called *bits*.

0000 0000 0011 0001 0010 0011 …

- These bits are grouped into *instructions*: very simple actions such as moving a number from one place to another.
- Good news: we don't have to deal with this soup, because we have high level languages with compilers & interpreters!

A Generic Compiler



Java Source Code

- *Java* is a high level programming language. Here's how programs are made with Java:
- 1) A person writes Java source code and saves it into a .java file. Example:

```
/* this is a simple Java program */
class Example {
  public static void main(String args[]) {
     System.out.println("this is a simple Java program");
   }
}
```

Java Byte Code

- 2) The Java compiler (named *javac*) compiles the .java file into a .class file containing *Java Byte Code*, an intermediate level of code somewhere **between** Java source code and machine code. Example:

```
public static void
main(java.lang.String[]);
Code:
0: iconst_0
1: istore_1
2: goto 30
5: getstatic
8: new
11: dup
12: ldc
14: invokespecial #23
17: iload_1
18: invokevirtual #27
21: invokevirtual #31
…
```

The Java Virtual Machine

- 3) The *Java Virtual Machine* (JVM), also called the *Java Runtime Environment* (JRE), runs the Java Byte Code.
- The JVM is an *interpreter**. It takes each line of Java Byte Code one at a time, turns it into machine code, and runs it.
  - Picture recap:

  - Contrast with a compiler.

* a half-truth

Why bother with all that?

- Compiling Java is complicated, so why is it done this way?
- Every type of CPU ever made has different machine code instructions. You can't learn them all!
- Java Byte Code is **universal**. You can run it on Microsoft Windows, Solaris, Linux, Mac OS, …
  - All you need is the Java Virtual Machine on your computer, and Oracle gives that away for free!

Processing

- In this course, we'll learn a high level language called *Processing*.
  - Download free for your Windows, Mac, or Linux computer at http://www.processing.org/
- It's very similar to Java.
  - Think of it as an extra "layer" of code on top of Java.
  - Processing source code gets turned into Java source code, and given to the Java compiler.
- Processing has a lot of built-in features that make pictures, drawings, and animations easy.
- Easy to transition to Java later this semester, and next semester in CSCI 202.

The Processing Window

Menu & Toolbar

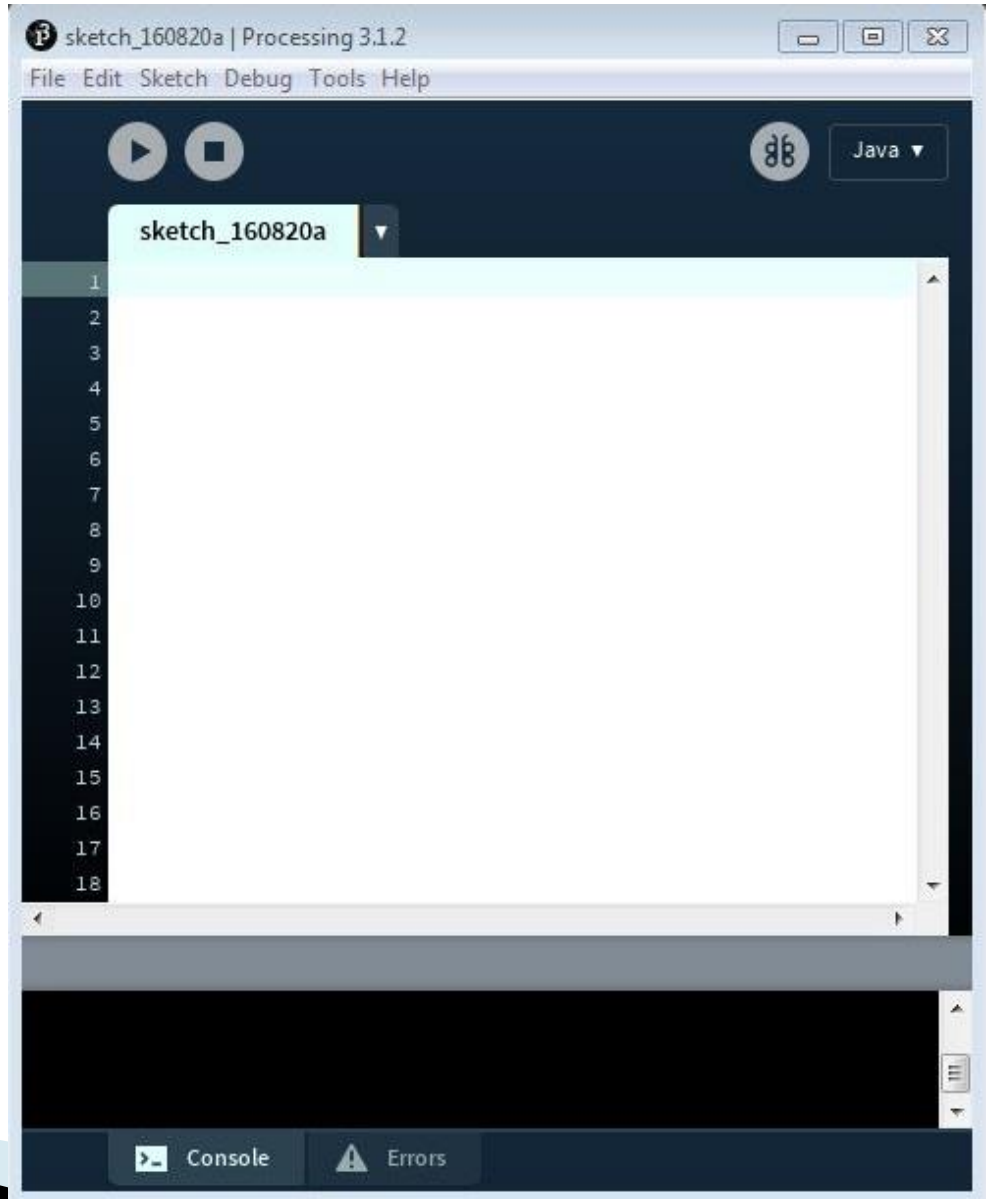
Run Button ⇨

Tabs

Text editor

Messages & errors

Console



Hello World

- Let's do a simple program.
- 1) In the text area, type

```
println("Hello, world!");
```

- 2) Press the Run button on the toolbar.
  - (It's a triangle pointing to the right.)
- Two things happen:
  - A message will appear in the console

    ```
    Hello, world!
    ```

  - A small square window will open
    - We'll call it the *display window.* It's where images and animations will appear later.

Debugging

- The *syntax* of a language describes the rules of how to write code.
  - It's like English grammar rules, but for computer programs.
- If you mistype something, or get the syntax wrong, the program will not run. These are called *syntax errors* or simply *bugs*.
- Example:

```
println("Hello, world!";
```

- *Debugging* is the art of fixing these bugs.