### **Programming Language**

Programming languages exist to tell the computer what to do. In general, there are three types of programming languages based complexity.

* High Level Languages - What humans program in.
* Assembly Language - High Level Languages are converted into Assembly Language first.
* Machine Code - Assembly Language are converted to Machine Code to finally run on the computer.

### **Types of High Languages**

High Level Languages are divided in to two categories:

* Compiled - C, C++, Pascal, Java (these converted to Assembly Languages first with a Compiler)
* Interpreted - Python, Ruby, Javascript (these are run with being compiled by an Interpreter)

There are trade-offs to using each. Compiled languages are faster, where as Interpreted languages are easier to use.

### **Hello World**

A program that simply prints hello world is traditionally the first program you write when learning a new language or technology.

Create a HelloWorld.java file containing

public class HelloWorld {

public static void main(String[] args) {

// Prints "Hello, World" to the terminal window.

System.out.println("Hello, World");

}

}

* Java program must have at least one **class**; name starts with capital by convention.
* The class name, HelloWorld, and filename must match. Java files have **.java** extension.
* A class can have several **methods**. The program starts executing at main method.
* The only **statement** in this program prints the hello message. Statements end with semicolon. Replace "Hello Word" with 2+3 to see what happens.
* **Comments** start with "//" and end at the end-of-line; they can also be between "/\*" and "\*/".
* **Blocks** are encapsulated in "{" and "}". Note the placement of these curly braces; this is the accepted convention for Java.

### **Numeric Literals**

A constant value that appears directly in the program, e.g. 34, 0.34, 0.0000034

### **Numeric Variables**

* Hold data
* They are typed, e.g. int, float, etc.
* Declared with type followed by name, e.g.

int length;

double rate;

Above, the type is int (or Integer) and the name (or Identifier) is length.

* Identifier are names of variables
* Declare, initialize and update variables
* Use "final" to declare constant variables

### **Naming Conventions**

* Capitalize class names, e.g. Box
* Camel Case variable and method names, e.g. printValueNow
* Constants are in all caps

### **Assignments**

* Of the form *variable* = *expression*
* Example: "area = len \* width"
* Assignment is itself an expression that evaluates to the assigned value, e.g. x = y = 1

### **Reading from Keyboard**

Example of code to read an integer:

**import java.util.Scanner;**

public class ReadInteger {

public static void main(String[] args) {

**int num;**

// initialize Scanner

**Scanner input = new Scanner(System.in);**

// Prompt for input and read integer

System.out.print("Enter Number: ");

num = input.**nextInt()**;

// Display what is read

System.out.print("The integer read was: ");

System.out.println(num);

}

}

* Uses a class [Scanner](http://docs.oracle.com/javase/7/docs/api/java/util/Scanner.html)
* [(Links to an external site.)](http://docs.oracle.com/javase/7/docs/api/java/util/Scanner.html)
* in package [java.util](https://docs.oracle.com/javase/7/docs/api/java/util/package-use.html)
* [(Links to an external site.)](https://docs.oracle.com/javase/7/docs/api/java/util/package-use.html)
* Uses a variable called num
* Initializes a Scanner object called input to read from keyboard
* Reads value using method called nextInt()
* Other methods are nextByte(), nextShort(), nextDouble(), etc

### **Numeric Operators**

* + (add), - (subtract), \* (multiply), / (divide), % (remainder)
* augmented assignements +=, -=, \*=, /=, %=
* (pre/post) increment and decrement (++, --)

### **Boolean Literals**

* **true** and **false** (like 10, 3.4, etc.)

### **Boolean Variables**

* Holds a value of true or false
* For example:

boolean isPositive;

isPositive = true;

### **Relational Operators**

* Compare two values
* These are <, <=, >, >=, ==, !=
* Consider the following, if num is 5:

| **Operator** | **Meaning** | **Example** | **Value** |
| --- | --- | --- | --- |
| < | less than | num < 0 | false |
| <= | less than or equal to | num <= 0 | false |
| > | greater than | num > 0 | true |
| >= | greater than or equal to | num >= 0 | true |
| == | equal to | num == 0 | false |
| != | no equal to | num != 0 | true |

### **Boolean Expression**

* We know Numeric Expressions like 3+2\*5
* Use relational operators to create Boolean Expressions like (3+2\*5) > 20 and num < 0.