# Lectura 3: **Comparing Popular Programming Languages**

Since the 1950s, computer scientists have devised thousands of programming languages. Many are obscure, perhaps created for a Ph.D. thesis and never heard of since. Others became popular for a while then faded due to lack of support or because they were limited to a particular computer system. Some are variants of existing languages, adding new features like parallelism- the ability to run many parts of a program on different computers in parallel.

- **Comparing Programming Languages**

There are several ways to compare computer Languages but for simplicity, we'll compare them by Compilation Method and Abstraction Level.

- **Compiling to Machine Code**

Some languages require programs to be transformed directly into Machine Code- the instructions that a CPU understands directly. This transformation process is called compilation. Assembly Language, C, C++, and Pascal are compiled languages.

- **Interpreted Languages**

Other languages are either Interpreted such as Basic, Actionscript, and Javascript, or a mixture of both being compiled to an intermediate language - this includes Java and C#.

An Interpreted language is processed at runtime. Every line is read, analyzed, and executed. Having to reprocess a line every time in a loop is what makes interpreted languages so slow. This overhead means that interpreted code runs between 5 - 10 times slower than compiled code. The interpreted languages like Basic or JavaScript are the slowest. Their advantage is not needing to be recompiled after changes and that is handy when you're learning to program.

Because compiled programs almost always run faster than interpreted, languages such as C and C++ tend to be the most popular for writing games. Java and C# both compile to an interpreted language which is very efficient. Because the Virtual Machine that interprets Java and the .NET framework that runs C# are heavily optimized, it's claimed that applications in those languages are as fast if not faster as compiled C++.

- **Level of Abstraction**

The other way to compare languages is level of abstraction. This indicates how close a particular language is to the hardware. Machine Code is the lowest level, with Assembly Language just above it. C++ is higher than C because C++ offers greater abstraction. Java and C# are higher than C++ because they compile to an intermediate language called bytecode.

How Languages Compare

Fast Compiled Languages

Assembly Language

C

C++

Pascal

C#

Java

Reasonably Fast Interpreted

Perl

PHP

Slow Interpreted

JavaScript

ActionScript

Basic

Machine Code is the instructions that a CPU executes. It's the only thing that a CPU can understand and execute. Interpreted languages need an application called an **Interpreter** that reads each line of the program source code and then 'runs' it.