Table of Contents

[Differences between Java, C and C++ 2](#_Toc64375853)

[1. Static and dynamic programming language 2](#_Toc64375854)

[2. Pre-Processor is required in C and C++, but not required in Java 2](#_Toc64375855)

[3. Platform Dependent VS Platform Independent 4](#_Toc64375856)

# Differences between Java, C and C++

## Static and dynamic programming language

As we know that there are two types of programming language:

Dynamically typed programming language

Statically typed Programming Language

C and C++ are static programming languages, but Java is a dynamic programming language.

**Static programming language**

If any programming language allows memory allocation for primitive data types at compilation time then that programming language is called as static programming language (examples C and C++).

In C and C++ applications memory will be allocated for primitive data types at compilation time and not at runtime (off course In C and C++, calloc () and malloc () functions allows dynamic memory allocation, but in general C and C++ applications are static programming language).

**Dynamic programming language**

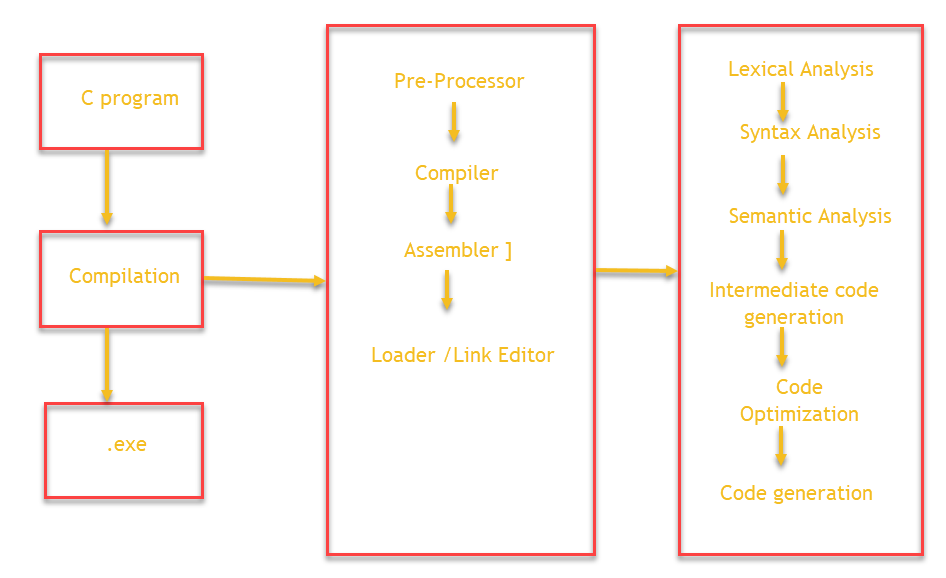
If any programming language allows memory allocation for primitive data types at run time then that programming language is called as dynamic programming language (example Java).

In Java applications, memory will be allocated for primitive data types at runtime only.

In Java memory is allocated for objects at runtime only, which means that memory is allocated for primitive types at run time only and not compilation time

## Pre-Processor is required in C and C++, but not required in Java

What is pre processor and where it is located.



Job of preprocessor:

In C, we use header files in application

#include <stdio.h>

#include <math.h>

Preprocessor will recognize all #include statements in the C++ application.

It will check take all header file names from #include

It will search for specified / required header files in C and C++ software’s,

if the specified header files are not present then error will be generated

if the specified header files exists then preprocessor will load all the required libraries.

Preprocessor will load all the specified header files to the memory.

Loading pre-defined libraries at compilation time is called static loading.

Conclusion: In C and C++, preprocessor is required to recognize #include statements in order to load header file content to the memory.

Why pre processor is not required in Java.

Compiler responsibilities

In java classes and interfaces are in the form on packages (like java.lang.\*, java.sql.\*, …)

We use import statements in Java like import java.lang.\*

When we compile java program, the job of the compiler is

Compiler will recognize all import statements

Compiler will take the specified packages from import statements

Compiler will go to Java software, where compiler will check whether these packages exists or not.

If packages do not exist then the compiler will raise an error

If packages exists then the compiler will not load the package content to memory.

When we execute Java application, job of JVM is

When the predefined classes and interfaces are identified by JVM, then JVM will load respective classes and interfaces into the memory.

In java applications, the predefined library is loaded by JVM as per requirement. This type of loading pre-defined libraries is called dynamic loading.

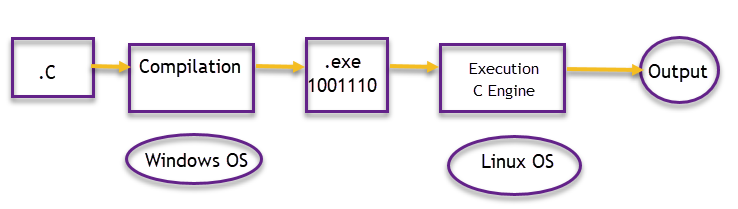
In java preprocessor is not required. In place of preprocessor, we have compiler and JVM.

<https://www.tutorialspoint.com/compiler_design/index.htm>

## Platform Dependent VS Platform Independent

C and C++ are platform dependent programming languages, but java is platform independent programming language

If any programming language allows its applications to perform compilation and execution on same operating systems, then that program is called is platform dependent language (example are Cand C++).



Program compiled in Windows OS: It generated .exe file (which contains sequence of 0’s and 1’s).

.Exe contains directly executable code and it has windows representation.

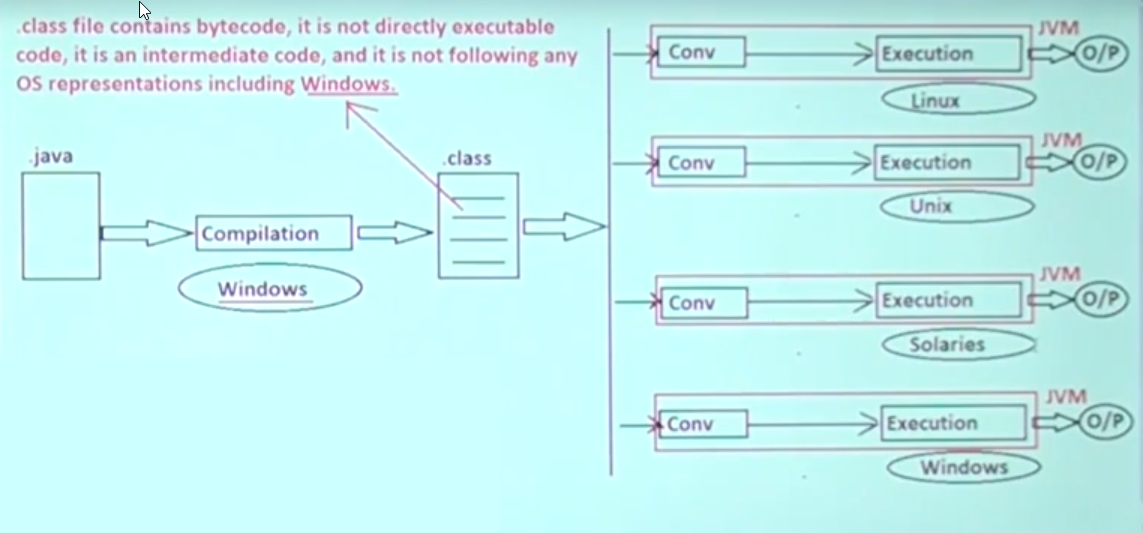
In above diagram c engine requires Linux representation code for execution and so we can’t execute windows representation.

So it becomes mandatory to compile and execute the code in same operating system in case of C and C++.

So how java becomes platform independent programming language?

If any programming language allows its applications to perform compilation and execution on other or same operating systems, then that program is called is platform dependent language (example Java).

The job of convertors and executors is to take intermediate (neutral) byte code and convert to OS executable code.



JVM (Java virtual machine) contains both conversion and execution mechanism. We have different JVM (JVM is platform dependent). Java software is platform dependent.

**Jvm is platform dependent** because we have different **JVM** for different operating system. **JVM** is one kind of interface or middleware between **OS**(Operating Systems) and java language. **JVM** provides the environment to execute the java file(. ... So **JVM is platform dependent.**

A class file contains the byte code generated from your original java source.

Byte code can only be interpreted by a JVM. An exe doesn’t need to be

interpreted since it's in machine code and the computer can read it directly.

|  |  |
| --- | --- |
| .exe | .class |
| It is in C and C++ | It is in Java |
| Contains executable code | Contains byte code (intermediate code). Intermediate code is executed by converters in JVM |
| Platform dependent | Platform independent |
| Less security (most viruses exists in form for .exe) | It is secured, as it don’t contain executable code |