Introduction to Java

Introduction

- Early computer languages such as Basic, Cobol, Fortran were not designed using structural principles.
- These languages relied upon GOTO statement as program control.
- So programs that are written in these languages are virtually impossible to understand.

Assembly Language

- It produces high efficient programs.
- But not easy to learn or understand.
- Also, difficult to debug.

C as a Programming language

- It is powerful, structured, efficient, easy to learn
- Priority is given to function rather than data
- Increasing complexity of programs.

Con't…

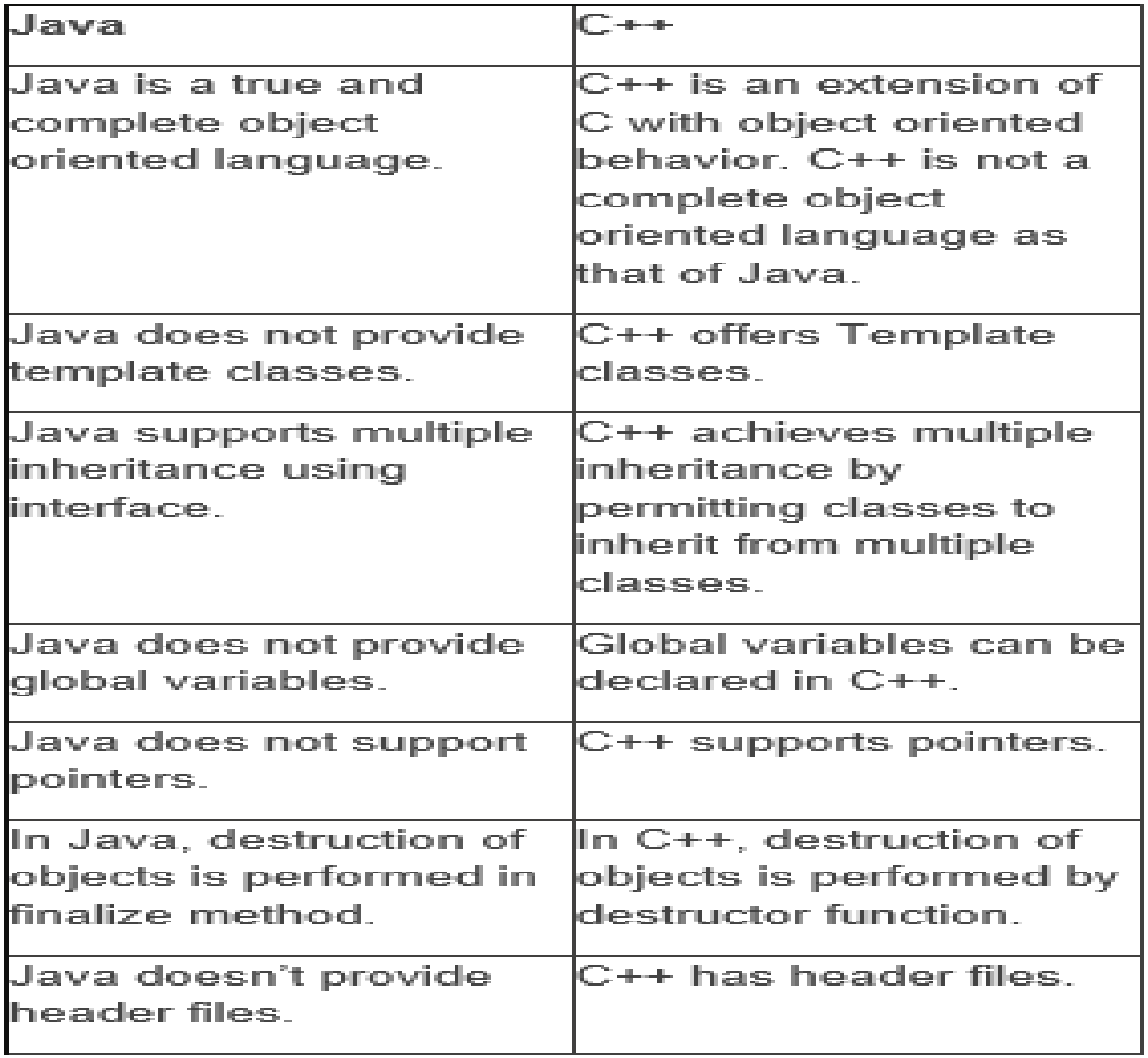
C++

- Provides the need for better ways to manage complexity.
- Priority is given to data than functions.
- It is based on object oriented concept.
- Complex programs can be organize through the use of inheritance, encapsulation and polymorphism.

Major Differences Between C And JAVA

- JAVA is Object-Oriented while C is process-oriented.
- Java is an Interpreted language while C is a compiled language.
- C is a low-level language while JAVA is a high-level language.
- C uses the top-down approach while JAVA uses the bottom-up approach.
- Pointer go *backstage* in JAVA while C requires explicit handling of pointers.
- The Behind-the-scenes Memory Management with JAVA & The User-Based Memory Management in C.
- JAVA supports Method Overloading while C does not support overloading at all.
- Unlike C, JAVA does not support Preprocessors.
- C uses the printf & scanf functions as its standard input & output while JAVA uses the System.out.print & System.in.read functions.
- Exception Handling in JAVA And the errors & crashes in C.

Major Differences Between C++ And JAVA



Java	C++
Java is a true and complete object oriented language.	C++ is an extension of C with object oriented behavior. C++ is not a complete object oriented language as that of Java.
Java does not provide template classes.	C++ offers Template classes.
Java supports multiple inheritance using interface.	C++ achieves multiple inheritance by permitting classes to inherit from multiple classes.
Java does not provide global variables.	Global variables can be declared in C++.
Java does not support pointers.	C++ supports pointers.
In Java, destruction of objects is performed in finalize method.	In C++, destruction of objects is performed by destructor function.
Java doesn't provide header files.	C++ has header files.

What is Java?

History of Java

- It is an object-oriented language developed by team lead by James Gosling at Sun microsystems in the mid 1990s.
  - First the project was code named “Green”, Intended for embedded systems in 1991.
  - Then, it was renamed as “Oak”. Later on they realized that Oak was the name of existing computer language, so they changed the name to “JAVA” in 1995.
  - Symbol: coffee cup
- Unlike C++, it was developed from scratch.
  - The syntax is very similar to C.

Java Version History

- Even though Java is not very old, there are several key versions to be aware of:
  - Java 1.0.2 - First stable version. Not very useful.
  - Java 1.1 (1997)
    - Security, Database connectivity (JDBC), Improved Performance
    - Most stable version 1.1.8
    - Unstable versions 1.1.4 and 1.1.5
  - Java 1.2 (1998)   **MAJOR CHANGES**
    - Addition of Swing GUI (mostly replaces AWT)
    - Improved Security
    - Enterprise computing
  - Java 1.3 (2000)
    - Many extended APIs added
    - Improved performance
  - Java 1.4 (2002)
    - Improved performance
    - Bug Fixes

Salient features of Java

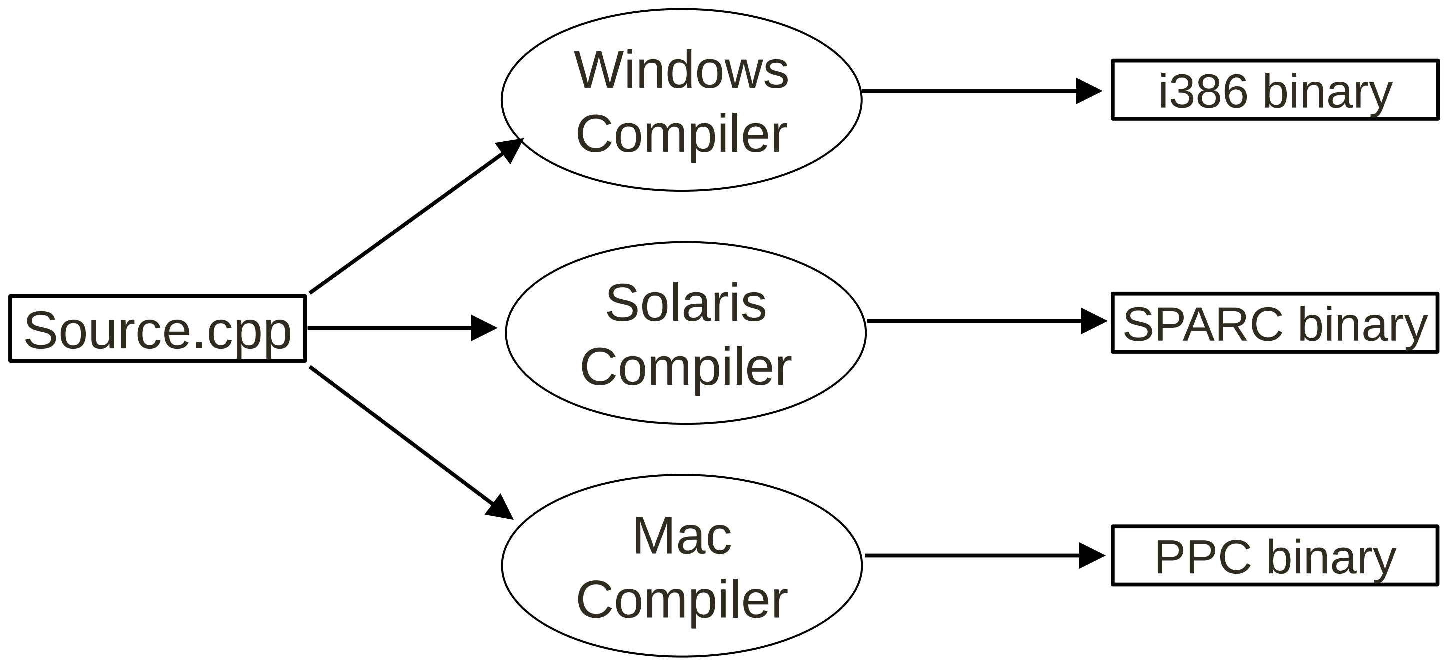
"A simple, object-oriented, distributed, interpreted, robust, secure, architecture neutral, portable, high-performance, multi-threaded and dynamic language."

Platform Independence

- Java has been described as WORA (Write once, Run Anywhere)
- Because Java source code is compiled to byte code and the byte code is interpreted, Java code can be executed anywhere an interpreter is available.
- The "Interpreter" is call the Java Virtual Machine.

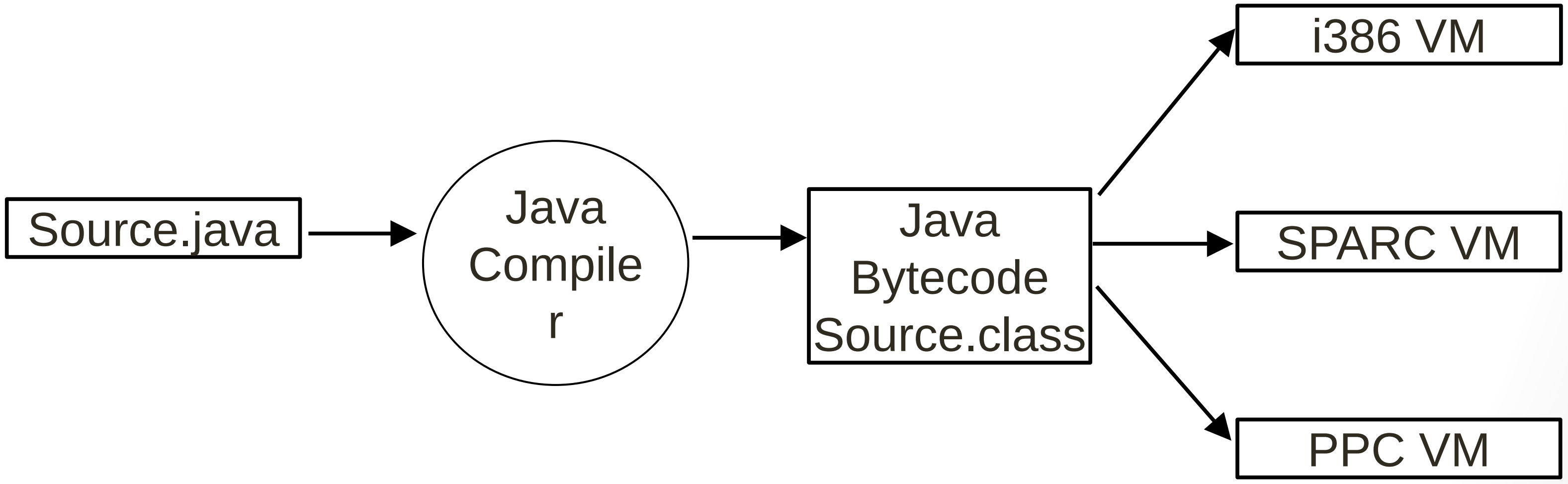
The Java Virtual Machine

- Traditionally, source code had to be compiled for the target hardware and OS platform:



The Java Virtual Machine

- Java source files (.java) are compiled to Java bytecode (.class)
- Bytecode is interpreted on the target platform within a Java Virtual Machine.
- Byte code remains same for all machines. So, it is platform independent.



Simple

- Similar to C/C++ in syntax
- But eliminates several complexities of
  - No operator overloading
  - No direct pointer manipulation or pointer arithmetic
  - No multiple inheritance
  - No malloc() and free() – handles memory automatically Garbage Collector
  - Does not use header files
- Lots more things which make Java more attractive.

Object-Oriented

Data abstraction

- Providing only essential information in program and hiding their background details.

Encapsulation

- The wrapping up of data and its functions into a single unit is called Encapsulation.

Inheritance

- It is the process of creating the new classes and using the behavior of the existing classes by extending them just to reuse the existing code and adding the additional features as needed.

Polymorphism

- Multiple form
- Polymorphism is the way of providing the different functionality by the functions having the same name based on the signatures of the methods.

Dynamic binding

- The process of linking procedure call to a specific sequence of code (method) at run-time.

Con’t

- Objective C, C++ fulfills the above characteristics yet they are not fully object oriented languages because they are structured as well as object oriented languages.
- But in case of java, it is a fully Object Oriented language because object is at the outer most level of data structure in java.
- No stand alone methods, constants, and variables are there in java.
- Everything in java is object even the primitive data types can also be converted into object by using the wrapper class.

Distributed

- Applications are constructed using objects. Objects can be distributed in multiple locations within a network environment.
- Extensive integration with TCP/IP.

Interpreted

- Java compiles to byte-code (not machine code). Byte code is interpreted.
- Most Java versions after 1.2 include a JIT (Just-In-Time) compiler.

Robust

- Java has the strong memory allocation and automatic garbage collection mechanism.
- It provides the powerful exception handling and type checking mechanism as compare to other programming languages.
- Compiler checks the program whether there is any error and interpreter checks any run time error and makes the system secure from crash.

Secure

- Java does not use memory pointers explicitly.
- Security manager determines the accessibility options of a class like reading and writing a file to the local disk.
- Java uses the public key encryption system to allow the java applications to transmit over the internet in the secure encrypted form.

Architecture-Neutral / Portable

- "Write-Once Run-Anywhere"
- Compiled Java (byte code) will run on any platform which has a Java Virtual Machine.
- The Java Virtual Machine is available for almost all platforms...
  - Even mainframes.

High-Performance

- Java uses native code usage, and lightweight process called threads.
- In the beginning interpretation of byte code resulted in slow performance but the advance version of JVM uses the adaptive and just in time compilation technique that improves the performance.

Multi-Threaded

- Multithreading means handling more than one job at a time.
- Processes contain multiple threads of execution.
- Similar to multi-tasking but all threads share the same memory space.

Dynamic

- Makes heavy use of dynamic memory allocation.
- Classes can be dynamically loaded at any time.

Components

Java have 4 components:

- Java Programming language : Program code - .java file.
- Java class file format : Compiling .java file which creates .class file(byte code)
- Java virtual machine : .class file is executed within JVM
- Java Application Programming Interface(Java API) : It contains set of pre-created libraries, available with Java s/w development kit(SDK) each of which holds various methods that can be accessed by any program on demand.
- JVM and Java API forms a platform for executing java programs. Together they are called Java Runtime Environment(JRE).
- Each java application runs inside a JVM. The JVM starts when a java program executes and terminates when the program ends.
- The no. of JVMs running on the machine is equal to the no. of java programs that are being executed.

The Java Technology

- Java Standard Edition (J2SE)- (PC development)
  - J2SE can be used to develop client-side standalone applications or applets.
- Java Enterprise Edition (J2EE) -(Distributed and Enterprise Computing)
  - J2EE can be used to develop server-side applications such as Java servlets and Java Server Pages.
- Java Micro Edition (J2ME)-(for handheld and portable devices)
  - J2ME can be used to develop applications for mobile devices such as cell phones.