Chapter - 1

The Way Java Works:

You'll type a source code file, compile it using the javac compiler, then run the compiled bytecode on a JVM. The JVM translates the bytecode into something that the underlying machine understands, and runs your program.

Source Code → Compiler → Output → Virtual machines
(code.java) (javac) (code.class) (Runs your programs)
(made up of bytecodes)
(platform independent)

Java Versions:

Java 1.1 → Java 2 → Java 5.0(Tiger)
(1.2, 1.3, 1.4) (Java 1.5)

Why does everything have to be in a class?
⇒ Java is an object-oriented language. A class is a blueprint for an object, and that nearly everything in java is an object.

Java can have dozens of classes but only 1 with a main method
⇒ that one that starts the program running.

A boolean and Integer are not compatible types in Java.
⇒ while(1) won't work in Java.

System.out.print Vs System.out.println
⇒ println is used for a newline

The Compiler Vs the JVM

Compiler Functions:

1. Java is a strongly typed language and the compiler can’t allow variables to hold data of the wrong type.
2. There are some datatype exceptions that may arrive during the runtime but the Compiler has to allow them in order to support some important functions of Java-Dynamic Binding.
3. At runtime Java may include some functions that weren’t even known to the programmer, so the compiler needs to be somewhat flexible. But its job is to stop anything that would never ever succeed at runtime.
4. Prevents access violation such as code trying to invoke a private method or change a method - for security reasons must never be changed.
5. Stops people from touching the code that they are not supposed to see, including code trying to access another class critical data.

JVM Functions:

1. To allow Java programs to run on any device or operating system (known as the “Write once, run anywhere” principle)
2. To manage and optimise program memory.