



CJ Unit 1

Java Introduction -

- In the year 1995, Sun Microsystems introduced a new programming language to the world - Java.
- Until then, the word 'Java' could only mean an island in Indonesia or a blend of coffee.
- Java is a language and a platform that helps professional programmers to develop wide range of applications on various platforms.
- Java is built upon C and C++. It derives its syntax from C and its features from C++.
- Java was initially developed to cater to small-scale problems, but was also found capable of addressing large-scale problems across the Internet. Very soon, Java was hugely successful and adopted by millions of programmers all across the world.
- JAVA is a truly object oriented programming language.

Java Features -

- Object Oriented
- High Performance
- Compiled & Interpreted
- Simple, Small and Familiar
- Platform Independent and Portable
- Multithreaded and Interactive
- Secure & Dynamic

Java Platform -

- A platform is a hardware or software environment in which a program runs.
- Some of the commonly used platforms are: Microsoft Windows, Linux, and Solaris.
- A number of these platforms, such as Linux and Solaris are a combination of operating system and underlying hardware components.
- The Java platform can be considered as an execution engine referred to as virtual engine and not a specific operating system or hardware.

Components of Java Architecture -

1. Java Development Kit (JDK) -

Java Development Kit is a Kit that provides the environment to **develop and execute (run)** the Java program. JDK is a kit (or package) that includes two things

- Development Tools (to provide an environment to develop your java programs)
- JRE (to execute your java program)

2. Java Runtime Environment (JRE) -

Java Runtime Environment is an installation package that provides an environment to **only run (not develop)** the java program(or application) onto your machine. JRE is only used by those who only want to run Java programs that are end-users of your system.

3. Java Virtual Machine (JVM) -

Java Virtual Machine is a very important part of both JDK and JRE because it is contained or inbuilt in both. Whatever Java program you run using JRE or JDK goes into JVM and JVM is responsible for executing the java program line by line, hence it is also known as an **interpreter**.

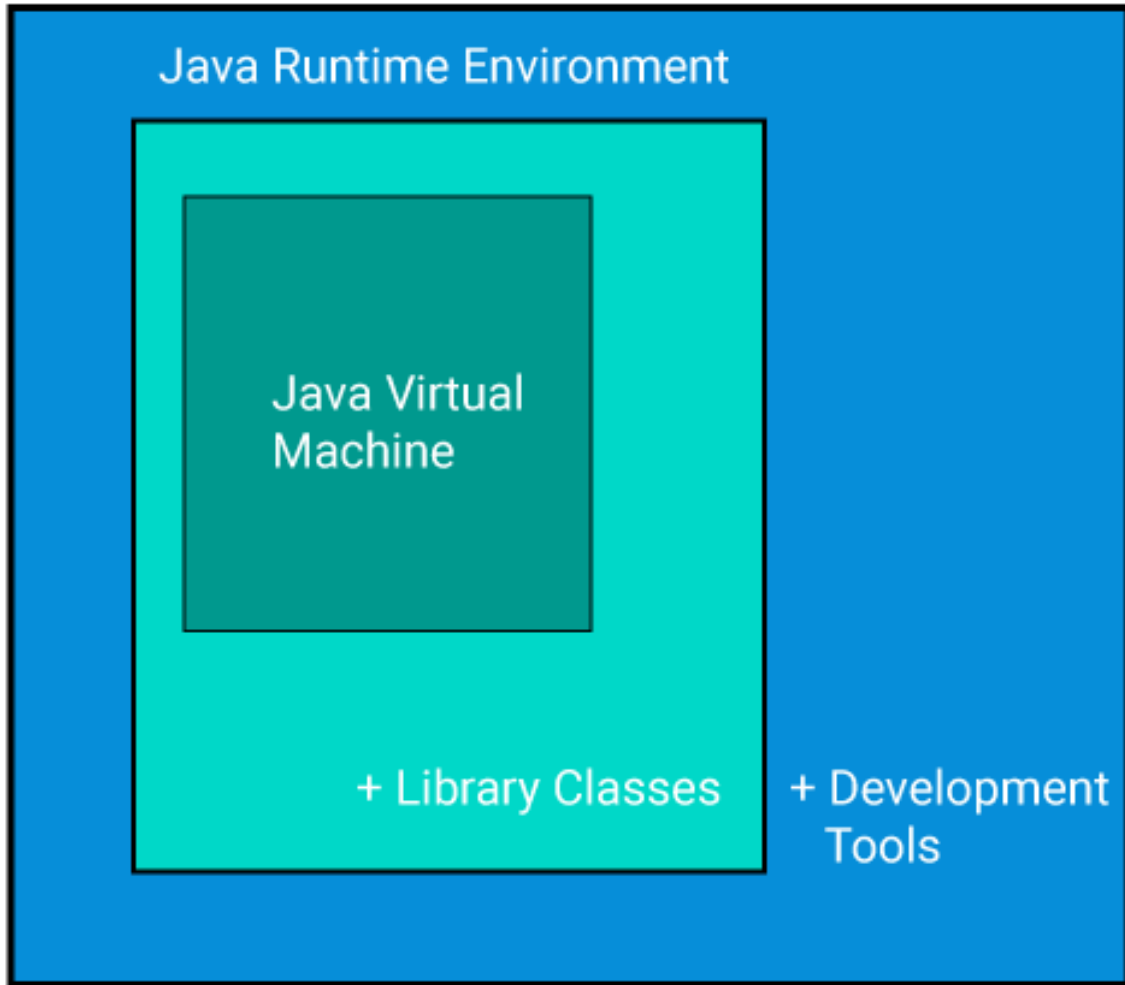
- The Java Virtual Machine (JVM) is the Java runtime environment and is available on different operating systems.
- It serves as the intermediary between a Java program and a host computer.

- JVM executes compiled Java programs (bytecodes). It forms a layer of abstraction for:
 - Underlying hardware platform
 - Operating system
 - Compiled code
- Different versions of JVMs are available for different operating systems.

Java Runtime Environment (JRE) -

- A software program needs to execute, and to do that it needs an environment to run in.
- It is a software layer that runs on top of a computer's operating system, providing additional services specific to Java.
- It loads class files and ensures there is access to memory and other system resources to run them.
- The JRE smoothens over the diversity of operating systems, ensuring that Java programs can run on virtually any OS without modification.

Components of JRE -



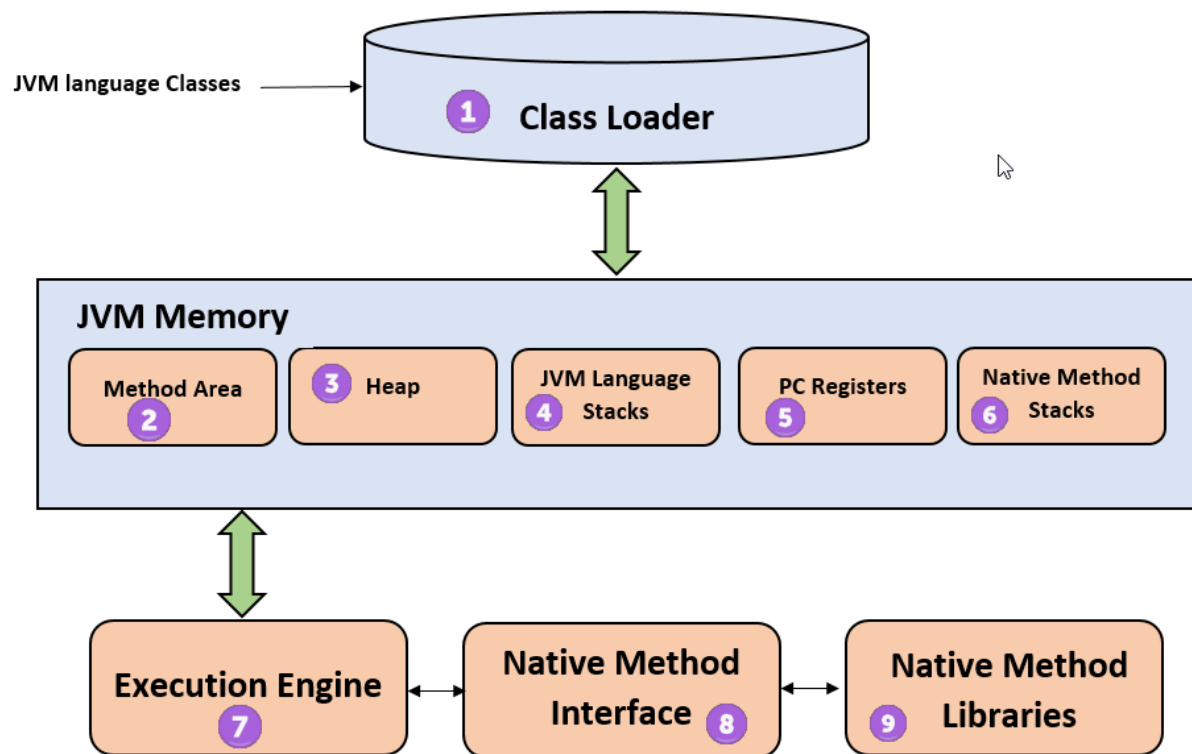
JDK = JRE + Development Tool

JRE = JVM + Library Classes

Java Virtual Machine (JVM) -

- Is a running software system responsible for executing live Java programs.
- Converts Java bytecode into machines language.
- JVM is a part of Java Run Environment (JRE).

Working of JVM -



Components of JVM -

1. CLASS LOADER -

The class loader is a subsystem used for loading class files. It performs three major functions viz. Loading, Linking, and Initialization.

2. METHOD AREA -

JVM Method Area stores class structures like metadata, the constant runtime pool, and the code for methods.

3. HEAP -

All the Objects, their related instance variables, and arrays are stored in the heap. This memory is common and shared across multiple threads.

4. JVM LANGUAGE STACKS -

Java language Stacks store local variables, and it's partial results. Each thread has its own JVM stack, created simultaneously as the thread is created.

5. PC REGISTERS -

PC register store the address of the Java virtual machine instruction which is currently executing. In Java, each thread has its separate PC register.

6. NATIVE METHOD STACKS -

Native method stacks hold the instruction of native code depends on the native library.

7. EXECUTE ENGINE -

- It is a type of software used to test hardware, software, or complete systems. The test execution engine never carries any information about the tested product.
- **JIT (Just In Time) compiler** - It helps in improving the performance of Java applications by compiling bytecodes to machine code at run time
- **Garbage collector** - It tracks each and every object available in the JVM heap space and removes unwanted ones.

8. NATIVE METHOD INTERFACE -

The Native Method Interface is a programming framework. It allows Java code which is running in a JVM to call by libraries and native applications.


9. NATIVE METHOD LIBRARIES -

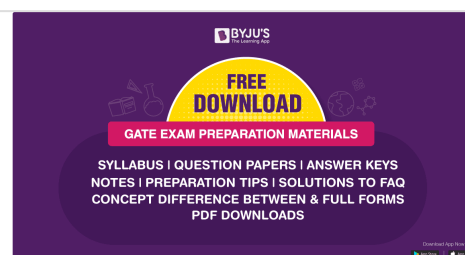
Native Libraries is a collection of the Native Libraries(C, C++) which are needed by the Execution Engine.

Difference between JDK,JRE, JVM -

Difference Between JDK, JRE, and JVM

Difference Between JDK, JRE, and JVM: JDK is a software development environment. JRE is a set of software tools designed for running other software. JVM is an abstract machine

 [https://byjus.com/gate/difference-between-jdk-jre-and-jvm/#:~:text=The%20JRE%20is%20an%20abbreviation,abbreviation%20for%20Java%20Virtual%20Machine.&text=The%20JDK%20\(Java%20Development%20Kit,JavaDoc%2C%20compilers%2C%20etc.\)](https://byjus.com/gate/difference-between-jdk-jre-and-jvm/#:~:text=The%20JRE%20is%20an%20abbreviation,abbreviation%20for%20Java%20Virtual%20Machine.&text=The%20JDK%20(Java%20Development%20Kit,JavaDoc%2C%20compilers%2C%20etc.))



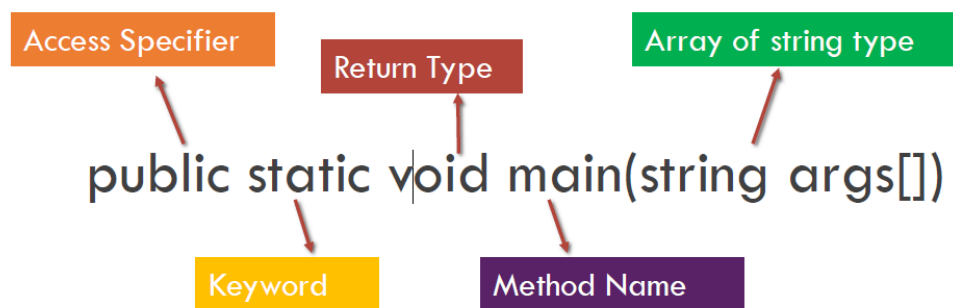
Java Program Structure -

- Java is Case Sensitive.
- The name you give to a source file is very important.
- The name of the class defined by the program should be the same as that of the source file. Java is Case Sensitive.

- The Java compiler requires that a source file use the .java filename extension.
- To compile the java program, execute the compiler, javac, followed by the name of the source file. ex. C:\>javac Example.java

Reason behind using these keywords -

Main Method



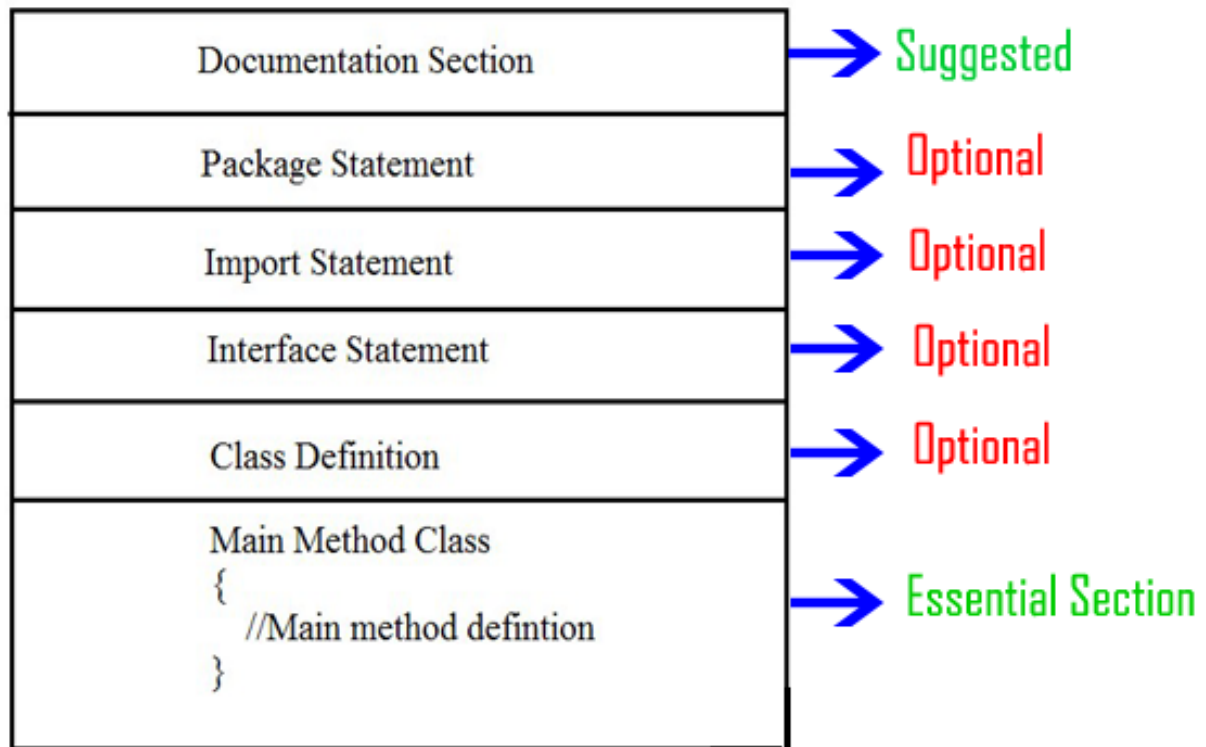
- **Public:** Use a public keyword before the main() method so that JVM can identify the execution point of the program.
- **Static:** We should call the main() method without creating an object.
- **Void:** Void keyword acknowledges the compiler that main() method does not return any value.
- **Main:** It is a default signature which is predefined in the JVM. It is called by JVM to execute a program line by line and end the execution after completion of this method.
- **String args[]:** The main() method also accepts some data from the user. It accepts a group of strings, which is called a string array. It is used to hold the command line arguments in the form of string values.

Simple program -

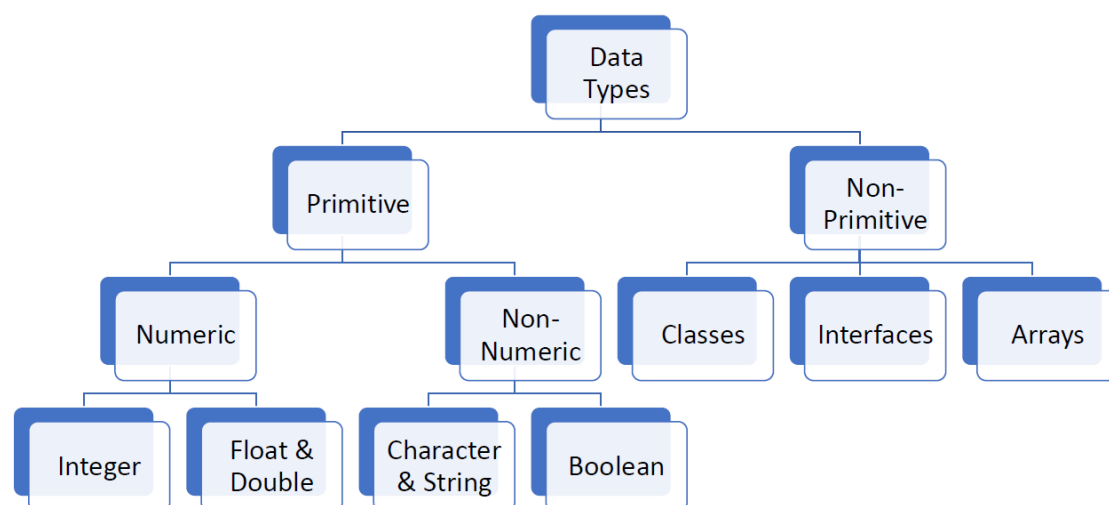
```
Example.java
class Example{
```

```
public static void main(String args[]){
System.out.print("The same old!!! HELLO")
}
```

General Program Structure -



Data Types -



The Primitive Types

Name	Width(Size in bits)	Range	Example
long	64	−9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	long a=2000000000;
int	32	−2,147,483,648 to 2,147,483,647	int b = 200000;
short	16	−32,768 to 32,767	short c = 2000;
byte	8	−128 to 127	byte d = 20;
float	32	1.4e−045 to 3.4e+0	float e = 2.7f;
double	64	4.9e−324 to 1.8e+308	double f = 25.326;
char	16	0 to 65,536	char g = 'x';
boolean	1	true or false	boolean h= true;

String Literals -

String literals in Java are specified by enclosing a sequence of characters between a pair of double quotes. ex. "This is a String!"

Escape Sequence -

Escape Sequence	Description
\'	Single quote
\"	Double quote
\\	Backslash
\r	Carriage return
\n	New line (also known as line feed)
\f	Form Feed
\t	Tab
\b	Backspace

Standard Default values -

Type of variable	Default value
Byte	zero
short	zero
int	zero
long	Zero: 0L
float	0.0f
double	0.0d
char	Null Character
boolean	false
reference	null

Scanner Class -

- The Scanner class is used to get user input.
- It is found in the java.util package.

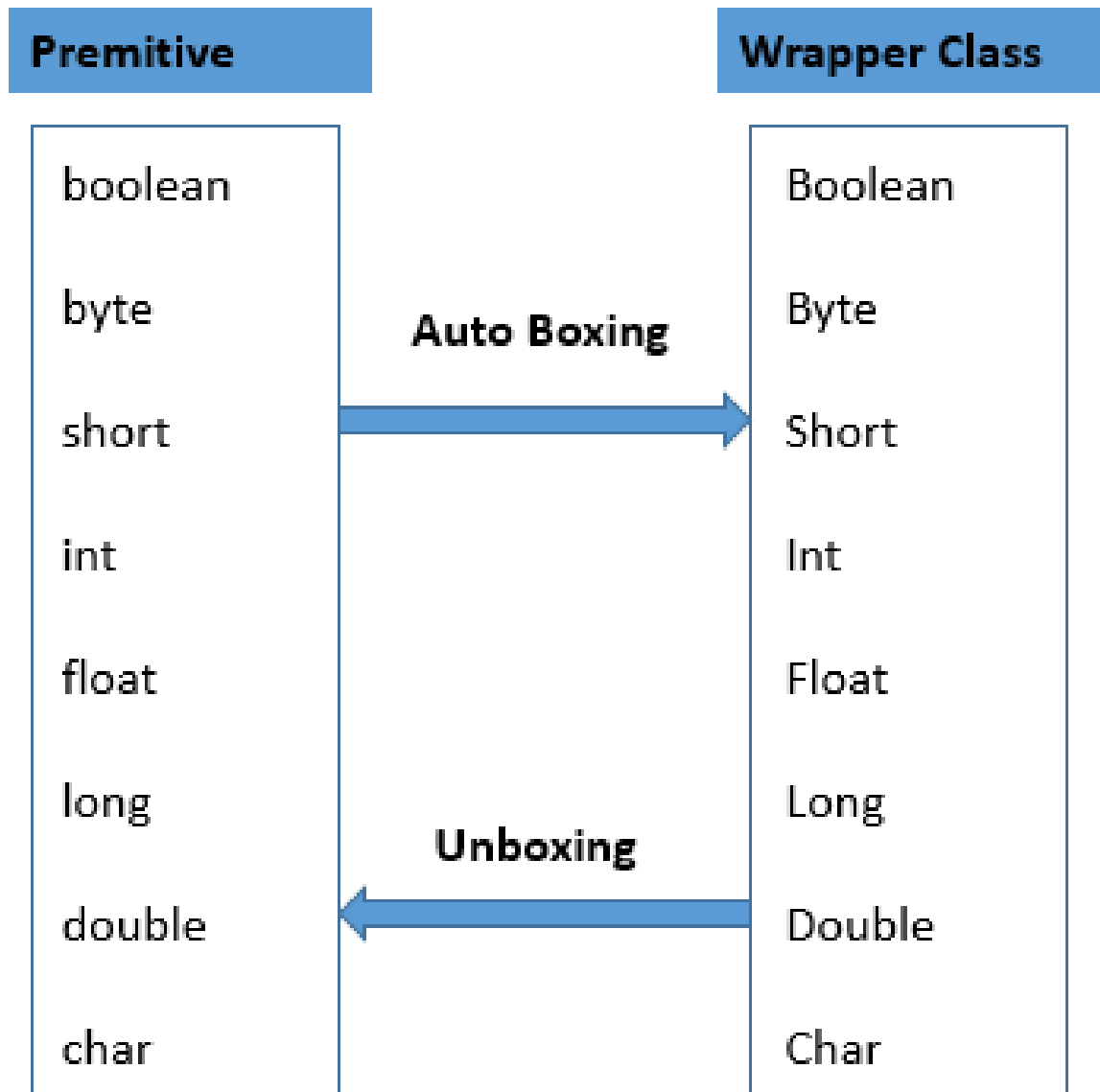
Method	Description
public String nextLine()	Reads a String value from the user
public byte nextByte()	Reads a byte value from the user
public short nextShort()	Reads a short value from the user
public int nextInt()	Reads a int value from the user
public long nextLong()	Reads a long value from the user
public float nextFloat()	Reads a float value from the user
public double nextDouble()	Reads a double value from the user
public boolean nextBoolean()	Reads a boolean value from the user

Type Conversion and Casting -

S.NO	TYPE CASTING	TYPE CONVERSION
1.	In type casting, a data type is converted into another data type by a programmer using casting operator.	Whereas in type conversion, a data type is converted into another data type by a compiler.
2.	Type casting can be applied to compatible data types as well as incompatible data types .	Whereas type conversion can only be applied to compatible datatypes .
3.	In type casting, casting operator is needed in order to cast the a data type to another data type.	Whereas in type conversion, there is no need for a casting operator.
4.	In typing casting, the destination data type may be smaller than the source data type, when converting the data type to another data type.	Whereas in type conversion, the destination data type can't be smaller than source data type.
5.	Type casting takes place during the program design by programmer.	Whereas type conversion is done at the compile time.
6.	Type casting is also called narrowing conversion because in this, the destination data type may be smaller than the source data type.	Whereas type conversion is also called widening conversion because in this, the destination data type can not be smaller than the source data type.
7.	Type casting is often used in coding and competitive programming works.	Whereas type conversion is less used in coding and competitive programming as it might cause incorrect answer.
8.	Type casting is more efficient and reliable.	Whereas type conversion is less efficient and less reliable.

Wrapper Classes -

- A Wrapper class is a class whose object wraps or contains primitive data types.
- Example -
Integer I = new Integer(25);



Autoboxing -

The automatic conversion of primitive data types into its equivalent Wrapper type is known as Autoboxing. (Primitive → Wrapper)

Ex. -

```
class Autoboxing{  
    public static void main(String[]  
    args  
    int a=50;  
    Integer a2=new Integer(a); //Boxing
```

```

Integer a3=5; //Boxing
System.out.println
(a2+" "+
}
}

```

Output - 50 5

Unboxing -

The automatic conversion of Wrapper type into its equivalent Primitive type is known as Unboxing.

(Wrapper → Primitive)

Ex. -

```

class Unboxing {
public static void main(String
args
Integer
i =new Integer(
int a=
i
System.out.println
(a);
}
}

```

Output - 50

Java Operators -

Arithmetic Operators -

Operator	Result
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus
++	Increment
--	Decrement
+=	Addition Assignment
-=	Subtraction Assignment
*=	Multiplication Assignment
/=	Division Assignment
%=	Modulus Assignment

Relational Operators -

Operator	Result
==	Equal to
>	Greater than
<	Less than
<=	Less than equal to
>=	Greater than equal to
!=	Not Equal to

Logical Operators -

Operator	Result
&&	Logical AND
	Logical OR
!	Logical Not

Bitwise Operators -

Operator	Result
~	Bitwise unary NOT
&	Bitwise AND
	Bitwise OR
^	Bitwise exclusive OR
>>	Shift Right
>>>	Shift right zero fill
<<	Shift left
&=	Bitwise AND assignment
!=	Bitwise OR assignment
^=	Bitwise exclusive OR assignment
>>=	Shift right assignment
>>>=	Shift right zero fill assignment
<<=	Shift left assignment

Conditional/Ternary Operators -

General Form: exp1 ? exp2 : exp3

