Type conversion in Java

1.What is Type Casting?

Type casting means **changing one data type into another data type**.

Example: converting int to double, or double to int.

2.Implicit Type Casting (Automatic)

Also called **Widening Casting**
Java does it automatically (you don't need to write anything)

Rule

Small type → Big type

byte → short → int → long → float → double

Example (Implicit Casting)

```
public class Main {
    public static void main(String[] args) {
        int a = 10;
        double b = a;   // automatic conversion

        System.out.println(b);
    }
}
```

Output:

```
10.0
```

Here int is converted to double automatically.

3.Explicit Type Casting (Manual)

Also called **Narrowing Casting**
You must write the type manually.

Rule

Big type → Small type
(Data loss can happen)

Example (Explicit Casting)

```
public class Main {
    public static void main(String[] args) {
        double x = 10.5;
        int y = (int) x;   // manual conversion

        System.out.println(y);
    }
}
```

Output:

```
10
```

.5 is lost (data loss).

Simple Real-Life Example

Implicit

Small bottle water poured into big bucket (no problem)

Explicit

Big bucket poured into small bottle (water will spill)

Type Casting	Meaning	Who does it	Example
Implicit	Small → Big	Java automatically	int → double
Explicit	Big → Small	Programmer manually	double → int

Implicit = safe conversion
Explicit = risky conversion (data loss)

4.What is Narrowing Conversion?

Narrowing conversion means:
Converting **bigger data type → smaller data type**

Data can be lost.

Why Called Narrowing?

Because we are going from **wide range → narrow range**.

Example:
 double (8 bytes) → int (4 bytes)

Narrowing Conversion Order

double → float → long → int → short → byte → char

5.Truncating Conversion

Truncating = Cutting off decimal part

When you convert **float/double → int**,
 Java **cuts (removes) the decimal part**.

Example of Truncating Conversion

```
public class Main {
    public static void main(String[] args) {
        double a = 9.99;
        int b = (int) a;   // truncating

        System.out.println(b);
    }
}
```

Output:

```
9
```

 .99 is removed (not rounded).

Important Point

- Java **does not round**, it only **cuts** decimal part.

6.Automatic Type Promotion

Automatic type promotion happens in **expressions** (during calculations).

Java automatically converts smaller types into bigger type.

Rule of Automatic Type Promotion

byte → short → char → int → long → float → double

In arithmetic operations, Java **promotes all to at least int**.

Example 1: byte + byte

```
public class Main {
   public static void main(String[] args) {
      byte a = 10;
      byte b = 20;
      byte c = (byte)(a + b);  // promotion happens

      System.out.println(c);
   }
}
```

Without casting:

```
byte c = a + b; // ERROR
```

Because a + b becomes **int automatically**.

Example 2: char + int

```
public class Main {
   public static void main(String[] args) {
      char ch = 'A';  // ASCII 65
      int x = 10;

      int result = ch + x;   // char promoted to int

      System.out.println(result);
   }
}
```

Output:

```
75
```

Example 3: int + double

```
public class Main {
    public static void main(String[] args) {
        int a = 10;
        double b = 5.5;

        double c = a + b;  // int promoted to double

        System.out.println(c);
    }
}
```

Output:

```
15.5
```

Trick to Remember

Truncating conversion

Decimal value is CUT
Happens in explicit casting

Automatic Type Promotion

Happens in expressions
Java converts small → big automatically

Truncating conversion: Removing fractional part during type casting.
Automatic type promotion: Java automatically converts smaller data types to larger types during expression evaluation.

```
byte a = 10;
byte b = 20;
byte c = a + b;   // WHY ERROR?
```

Because a + b becomes **int by automatic promotion**.