🔷 Implicit Casting (Widening Casting)

- This happens **automatically** when converting from a **smaller data type to a larger data type**.
- ✅ **Safe** – no data is lost.

📌 Example:

```
short s = 10; int i = s; // short to int double d = i; // int to double
System.out.println(d); // 10.0
```

Java automatically does this because the larger type can hold all possible values of the smaller type.

🔶 Explicit Casting (Narrowing Casting)

- You need to do this **manually** when converting from a **larger data type to a smaller data type**.
- ⚠️ **Not safe** – you might lose data or precision.

📌 Example:

```
double d = 10.5; int i = (int) d; // double to int, explicit cast System.out.println(i);
// 10 (decimal part is lost)
```

📋 Widening Order (Lower to Higher):

```
byte → short → int → long → float → double
```

So all of these are **implicit** conversions.

📋 Narrowing Examples (Higher to Lower):

```
int i = 100; short s = (short) i; // explicit byte b = (byte) s; // explicit
```

💡 Summary Table:

From → To	Implicit/Explicit	Example
`int → double`	Implicit	`double d = i;`

From → To	Implicit/Explicit	Example
`double` → `int`	Explicit	`int i = (int) d;`
`short` → `int`	Implicit	`int i = s;`
`int` → `short`	Explicit	`short s = (short) i;`