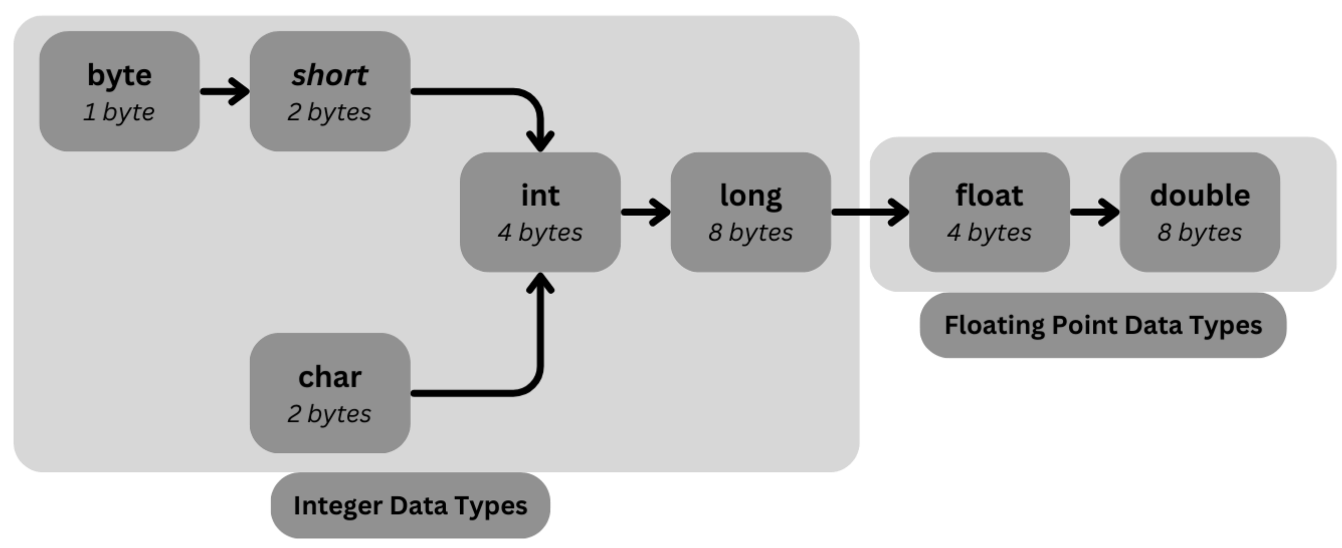
Primitive Type Casting

- Converting one type of data value into another type is known as **Type Casting**.
- This is specifically referred to as *Primitive Type Casting*.
- Applicable to all primitive data types except *boolean*.

Types of Type Casting

There are two types of data type casting:

- **Implicit Type Casting** (*Up-casting*, *Widening*)
- **Explicit Type Casting** (*Down-casting*, *Narrowing*)



Implicit Type Casting (Up-casting)

- Conversion of a *lower* datatype value into a *higher* datatype value.
- Handled automatically by the compiler.

```java
class UpcastingExample {
    public static void main(String[] args) {
        int intValue = 'A';
        System.out.println(intValue); // 65

        int intVal = 10;
        long longVal = intVal;
        System.out.println(longVal); // 10

        double doubleVal = 10;
        System.out.println(doubleVal); // 10.0

        float floatVal = 1.5f;
        double doubleVal2 = floatVal;
        System.out.println(doubleVal2); // 1.5

        byte byteVal = 10;
        int intVal2 = byteVal;
        System.out.println(intVal2); // 10
    }
}
```

Explicit Type Casting (Down-casting)

- Conversion of a *higher* datatype value into a *lower* datatype value.
- Must be explicitly specified by the programmer.

```
class DowncastingExample {
    public static void main(String[] args) {
        // int x = 10.5; // Error: cannot convert from double to int implicitly

        int intVal = (int) 10.5;
        System.out.println(intVal); // 10

        short shortVal = (short) 10.5;
        System.out.println(shortVal); // 10

        double doubleVal = 10.5;
        int intVal2 = (int) doubleVal;
        System.out.println(intVal2); // 10

        // int a = 1.5f; // Error: float to int not allowed implicitly

        long longVal = (long) 1.5f;
        System.out.println(longVal); // 1

        char ch = 3533;
        System.out.println(ch); // '?' (Unicode character)
```

```java

        // char ch2 = 65383; // Error: exceeds valid char range
    }
}
```

Miscellaneous Behavior

```java
class TypeCastingEdgeCases {
    public static void main(String[] args) {
        int intVal = 10;
        byte byteVal = 10;

        intVal = intVal + 1;
        // byteVal = byteVal + 1; // Error: result is int

        byteVal = (byte) (byteVal + 1);

        System.out.println(intVal); // 11
        System.out.println(byteVal); // 11
    }
}
```

Notes:

- `y + 1` results in `int` because `y` is `byte` and `1` is `int` → cannot assign directly to `byte`.
- `x++` is equivalent to `x = x + 1`.
- `y++` behaves differently: equivalent to `y = (byte)(y + 1)` due to type promotion rules.