03-Type conversion & casting

Type Conversion (Implicit Conversion)

Type conversion, also known as implicit conversion or widening conversion, occurs automatically when:

- A smaller data type is assigned to a larger data type.
- The conversion is between compatible data types.

Characteristics:

- Performed automatically by the compiler
- No data loss occurs.
- The destination type is always larger than the source type.

Example:

When storing data, we use variables of various data types, such as boolean, int, float, byte, or short. But can we change the type of a variable? Let's explore this with an example:

byte b = 127;

int a = 256;

b = a; // This line will cause a compilation error.

In the above example, we're trying to assign an int value to a byte variable. Since int can hold a larger range of values than byte, the compiler will generate an error because this assignment may lead to data loss.

However, if we reverse the assignment:

int a = b;

This is allowed because we're storing a byte value in an int variable, which has a larger range. This type of conversion, where a smaller type is converted to a larger type, is called **implicit type conversion** or **widening**. The compiler handles these conversions automatically.

TELUSKO

Type Casting (Explicit Conversion)

Type casting, also known as explicit conversion or narrowing conversion, is performed manually by the programmer when:

- A larger data type needs to be assigned to a smaller data type.
- The conversion may result in data loss.

Characteristics:

- requires explicit use of the casting operator
- Can be applied to both compatible and incompatible data types.
- May result in data loss or unexpected results

Code Example:

```
public class TypeCastingExample {
    public static void main(String[] args) {
        byte b = 127;  // Step 1
        int a = 257;   // Step 2
        b = (byte) a;  // Step 3

        System.out.println("Value of b after casting: " + b);  // Step 4
    }
}
```

Step 1: Declare and initialize a byte variable
byte b = 127;

b is initialized with the maximum value a byte can hold, which is 127.

Step 2: Declare and initialize an int variable
int a = 257;

a is an int variable initialized with 257.

Step 3: Cast int to byte using modulo operation
b = (byte) a;

Here, you assign a to b by casting.

The byte data type has a range from -128 to 127 (which is a total of 256 different values).

TELUSKO

Now, when you cast a to a byte, Java calculates the equivalent value within the byte range using the modulo operation:

This explicit conversion is known as **casting** or **narrowing**, where we manually convert a larger data type to a smaller one.

Modulo Calculation:

257 % 256 = 1

After complete code execution, our output window displays:

Output:

```
Value of b after casting: 1
```

Automatic promotion in **Java** is a feature that automatically converts smaller data types to larger data types when they are used in expressions or method calls. This conversion ensures that all operands in an expression or arguments in a method call are of compatible types, allowing the operation to be performed successfully.

Type Promotion:

Type promotion refers to the process where a smaller data type is automatically converted to a larger data type. For instance, an `int` can be promoted to a `long`, `float`, `double`, and so on. The JVM performs this automatic type promotion when a method that requires a larger data type is called with a smaller data type argument.

TELUSKO

```java
public class hello{
    public static void main(String[] args) {

        byte a=10;
        byte b=30;

        int result=a*b;

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

```
E:\Telusko>javac hello.java

E:\Telusko>java hello
300
```

Code explanation:

The program calculates the product of a and b using a * b. Although both a and b are byte types, the result is automatically promoted to an int type because in Java, arithmetic operations involving bytes result in an integer value. This is why the result is stored in an int variable named result.

TELUSKO