Type Conversion and Casting

Type Conversion and Casting

- To assign a value of one type to a variable of another type.
- Automatic Conversions and Casting Incompatible Types

➢ **Automatic Conversions (or widening conversion):**

- an automatic type conversion will take place if the following two conditions are met:
  I. The two types are compatible.
  II. The destination type is larger than the source type.
- When these two conditions are met, a widening conversion takes place.

Type Conversion and Casting

➢ **Automatic Conversions:**

- For widening conversions, the numeric types, including **integer** and **floating-point** types, are compatible with each other.
- However, there are no automatic conversions from the numeric types to **boolean**.
- Also, **char** and **boolean** are not compatible with each other.
- Java also performs an automatic type conversion when storing a literal integer constant into variables of type **byte, short, long,** or **char**.

Type Conversion and Casting

➢ **Casting Incompatible Types (or narrowing conversion):**

- not all type conversions are implicitly allowed.
- you must use a cast, which performs an explicit conversion between incompatible types.
- you are explicitly making the value narrower so that it will fit into the target type.
- To create a conversion between two incompatible types, you must use a cast. A cast is simply an explicit type conversion.

```
(target-type) value
```

For instance,

```
int a;
byte b;
b = (byte) a;
```

Type Conversion and Casting

➢ **Casting Incompatible Types (or narrowing conversion):**

- if the size of the whole number component is too large to fit into the target integer type, then that value will be reduced modulo the target type's range.
- A different type of conversion will occur when a floating-point value is assigned to an integer type (**truncation**).
- Because, integers do not have fractional components.
- When a floating-point value is assigned to an integer type, the fractional component is lost.

Type Conversion and Casting

➢ Automatic Type Promotion in Expressions:

- In an expression, the precision required of an intermediate value will sometimes exceed the range of either operand.

  For example, examine the following expression:

  byte b = 50;

  b = b * 2;

- subexpression b*2 is performed using integers—not bytes.

Type Conversion and Casting

➢ **The Type Promotion Rules:**

- Java defines several type promotion rules that apply to expressions.
- First, all **byte, short,** and **char** values are promoted to **int**, as just described.
- If one operand is a **long**, the whole expression is promoted to **long**.
- If one operand is a **float**, the entire expression is promoted to **float**.
- If any of the operands are **double**, the result is **double**.