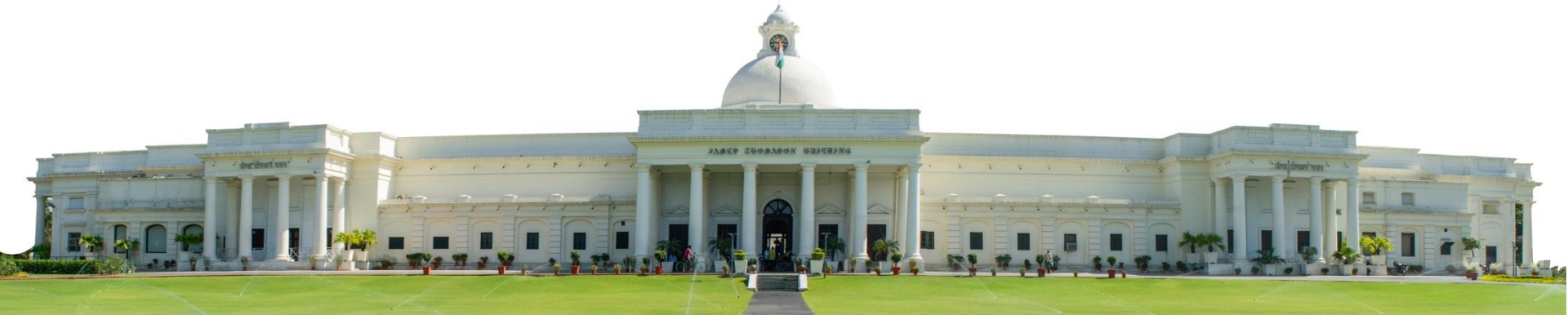
CSN-103: Fundamentals of Object Oriented Programming

Instructor: Dr. Rahul Thakur

Assistant Professor, Computer Science and Engineering, IIT Roorkee



Type Conversion and Casting

- In programming, it is fairly common to assign a value of one type to a variable of another type
- In Java, if two types are compatible, automatic conversion is performed
  - For example, int value to a long variable
- For incompatible data types, explicit conversion is required.
  - For example, double to byte
  - Explicit conversion is done using a ***cast***

Type Conversion and Casting

- Automatic type conversion conditions:
  - Two types are compatible
  - Destination type is larger than the source
    - **Widening conversion** happens
- For **Widening conversion,** numeric types (integer and floating-point) are compatible with each other
- Numeric types are not automatically converted to **char** or **Boolean**
- **char** and **boolean** are not compatible with each other

- Automatic type conversion also happens when storing an integer constant to **byte**, **short**, or **long** type

Type Conversion and Casting

- Explicit type conversion:
  - Two types are not compatible
  - Destination type is smaller than the source
    - **Narrowing conversion** happens
- **Cast**: An explicit type conversion

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

Type Casting

Type Casting not required

Type Conversion and Casting

- Problem with explicit type conversion: Loss of information
- Example: floating-point → integer
  - Truncation happens: Only integer part is stored, fraction is lost

```
int i = 250.667;          //Error
int i = (int) 250.667;    // i will store only 250
```

- Example: Target type has smaller range

```
int i = 128;
byte b = (byte) i;
```

Automatic Type Promotion

- Apart from Assignment, type conversion also occurs in **Expressions**
- Example

```
byte a = 40;
byte b = 50;
byte c = 100;
int d = a * b / c;
```

The result of a * b exceeds the range of byte. Java **automatically promotes** byte (and short) to int while evaluating an expression

Automatic Type Promotion

- Automatic promotion can cause compile-time errors

```
byte b = 50;
b = b * 2;          // Value assigned fits the target size
```

error: incompatible types: possible lossy conversion from int to byte

```
byte b = 50;
b = (byte) (b * 2);          // Will yield the correct value of 100
```

Note: b = (byte) b * 2; won't work…why??

Type Promotion Rules

- **byte** and **short** are promoted to **int**
- If one operand is **long**, the whole expression is promoted to **long**
- If one operand is **float**, the whole expression is promoted to **float**
- If one operand is **double**, the whole expression is promoted to **double**
- **Example:**

```
byte b = 42;
char c = 'a';
short s = 1024;
int i = 50000;
float f = 5.67f;
double d = .1234;
double result = (f * b) + (i / c) - (d * s);
```