ESc101 : Fundamental of Computing

I Semester 2008-09

Lecture 9+10

Types

- Range of numeric types and literals (constants) in JAVA
- Expressions with values of mixed types

I : Range of numeric types and literals (constants) in JAVA

(Primitive) Data types in JAVA

Domain	Java type
Integer	`byte, short, int, long`
Floating points (fractional numbers)	`float, double`
Boolean	`boolean`
Characters	`char`

Numeric types in JAVA

Type	range	size in bits
`byte`	-128 to 127	8
`short`	-32768 to 32767	16
`int`	-2×10^9 to 2×10^9	32
`long`	-9×10^{18} to 9×10^{18}	64
`float`	$\pm 3.4 \times 10^{38}$ and as small as $\pm 1.4 \times 10^{-45}$	32
`double`	$\pm 1.7 \times 10^{308}$ and more precision than `float`	64

Why different numeric types in JAVA

- arithmetic operations on floating points are more complex than on integers.
- Trade off between no. of bits and range(and/or precision)

Literals/Constants in Java

Definition : The fixed values that are presented in their human readable form.

We also call them *constants*.

`Examples :`

- the number `100` is an integer constant.
- the number `12.453` is a floating-point constant.
- `'a'` is a character constant.
- `true` is a Boolean constant.

What is Java type of a literal ?

Integer constants in java

- By default, integer constants are of type `int`.
- To specify a `long` constant append an **l** or **L**.

Example :

12 : `int`

12L : `long`

Assigning integer constants to integer data types in java

IMPORTANT RULE :

If `i` is a variable with integer data type, then we can assign any integer literal to `i` provided

the value of the literal is lying within the range of type of `i`*.*

Assigning integer constants to integer data types in java

Assignment	Valid or Invalid
`byte b = 12;`	??
`byte c = 156;`	??
`short s1 = 3245;`	??
`short s2 = 45678;`	??

Assigning integer constants to integer data types in java

Assignment	Valid or Invalid
`byte b = 12;`	valid
`byte c = 156;`	invalid
`short s1 = 3245;`	valid
`short s2 = 45678;`	invalid

Assigning integer constants to integer data types in java

Assignment	Valid or Invalid
`int i = 123;`	??
`int j = 3334445556667;`	??
`long l = 123;`	??
`long m = 3334445556667;`	??
`long n = 3334445556667L;`	??

Assigning integer constants to integer data types in java

Assignment	Valid or Invalid
`int i = 123;`	valid
`int j = 3334445556667;`	invalid
`long l = 123;`	valid
`long m = 3334445556667;`	invalid
`long n = 3334445556667L;`	valid

Floating point constants in Java

- By default, floating point constant are of type `double`.
- To specify a `float` constant, one must append an **f** or **F**.

`Examples :`

1.223 : `double`

12.3f : `float`

How to assign floating point literals to variables ?

1. float f = 13.4f; ??
2. float f = 2.3; ??
3. float f = 2; ??

Floating point constants in Java

- By default, floating point constant are of type `double`.
- To specify a `float` constant, one must append an **f** or **F**.

`Examples :`

1.223 : `double`

12.3f : `float`

How to assign floating point literals to variables ?

1. float f = 13.4f; **valid**
2. float f = 2.3; **invalid**
3. float f = 2; **valid**

Note : the reason for the validity/invalidity of the last two statements will be covered in next class when we discuss *type conversion during assignment*.

The order among the numeric types

Type	range
`byte`	-128 to 127
`short`	-32768 to 32767
`int`	-2×10^9 to 2×10^9
`long`	-9×10^{18} to 9×10^{18}
`float`	$\pm 3.4 \times 10^{38}$ and as small as $\pm 1.4 \times 10^{-45}$
`double`	$\pm 1.7 \times 10^{308}$ and more precision than `float`

The numeric types in the increasing order of their range

Based on the magnitude of the largest numeric value which can be stored in a data type (see last slide), the numeric types can be arranged from left to right in the increasing order of their range.

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

Terminology :

`int` is **wider** than `short` but **narrower** than **long**

`float` is **wider** than `long` but **narrower** than **double**

II : Expressions with values of mixed types

Expression built using different numeric types

E : expression with operands of different types?

How do we evaluate E ?

What is the type of E ?

Example : $1/2.3 * 4$

Evaluation of an expression

```
int i=2;   byte b=45;    double d = 34.567;
```

(we use different colors to differentiate between different types

$$\texttt{i} * \texttt{b} + \texttt{d}/\texttt{i} - \texttt{b}$$

Step 1 : (parenthesize):

$$\downarrow$$

$$(((\texttt{i} * \texttt{b}) + (\texttt{d}/\texttt{i})) - \texttt{b})$$

Step 2 :(replace the variables by their values)

$$\downarrow$$

$$(((2 * 45) + (34.567/2)) - 45)$$

Now we need to give rules for evaluation an expression of type $value_1 \; o \; value_2$.

Evaluation of an expression with single binary operator

E : $value_1\ o\ value_2$

o : a binary arithmetic operator.

`type`($value_1$) $\neq$ `type`($value_2$).

Evaluation of E is done in three steps.

1. $value_1$ and $value_2$ get promoted to `int` if they are of type **byte** or **short**.

$\downarrow$

2. if `type`($value_1$) is **wider** than `type`($value_2$) :

   $\Rightarrow$: type of $value_2$ gets **promoted** to `type`($value_1$).

   if `type`($value_2$) is **wider** than `type`($value_1$):

   $\Rightarrow$: type of $value_1$ gets **promoted** to `type`($value_2$).

   (At this stage `type`($value_1$) $=$ `type`($value_2$))

$\downarrow$

3. the expression is evaluated.

Evaluation of an expression with single binary operator

```
byte b=45;      double d = 34.567;
```

Expression : `b/d`

`b/d`

↓ (replacing the variables by their values)

`45/34.567`

↓ (promotion of `byte` to `int`)

`45/34.567`

↓ (promotion of `int` to `double`)

`45.0/34.567`

↓ (Evaluation)

`1.3018`