Java
Numeric Types and Operations (NUM)

Mario Alviano

University of Calabria, Italy

A.Y. 2016/2017

Detect or prevent integer overflow

- ✗ Underestimation of overflow exceptions is a common mistake
- ✓ Check critical operations
- ✓ Promote variables to larger domains during critical operations

```
https:
//www.securecoding.cert.org/confluence/display/
java/NUM00-J.+Detect+or+prevent+integer+overflow
```

Do not perform bitwise and arithmetic operations on the same data

- ✗ Don’t pretend to be better of a compiler
- ✓ Prefer readability
- ✓ Use bitwise operations on bit arrays

```
https://www.securecoding.cert.org/confluence/
display/java/NUM01-J.+Do+not+perform+bitwise+and+
arithmetic+operations+on+the+same+data
```

Ensure that division and remainder operations do not result in divide-by-zero errors

- ✗ Integer division by 0 is not defined
- ✗ Similar for integer division remainder
- ✓ Check the divisor

```
https:
//www.securecoding.cert.org/confluence/display/
java/NUM02-J.+Ensure+that+division+and+remainder+
operations+do+not+result+in+divide-by-zero+errors
```

Use integer types that can fully represent the possible range of unsigned data

- ✗ Java essentially does not support unsigned data
- ✓ Store unsigned values into type of larger domain
- ✓ Add a mask of bits to discard extra bits

```
https:
//www.securecoding.cert.org/confluence/display/
java/NUM03-J.+Use+integer+types+that+can+fully+
represent+the+possible+range+of++unsigned+data
```

Do not use floating-point numbers if precise computation is required

- ✗ Floating-point numbers approximate real numbers
- ✓ Prefer integers when possible
- ✓ Use `BigDecimal`

```
https://www.securecoding.cert.org/confluence/
display/java/NUM04-J.+Do+not+use+floating-point+
numbers+if+precise+computation+is+required
```

Do not attempt comparisons with NaN

- ✗ Comparator operators are messy for NaN
- ✓ Use `isNaN()`

```
https:
//www.securecoding.cert.org/confluence/display/
java/NUM07-J.+Do+not+attempt+comparisons+with+NaN
```

Check floating-point inputs for exceptional values

- ✗ Operations on floating-point numbers may result into infinity
- ✓ You should check for these cases

```
https://www.securecoding.cert.org/confluence/
display/java/NUM08-J.+Check+floating-point+inputs+
for+exceptional+values
```

Do not use floating-point variables as loop counters

- ✗ Floating-point variables are not precise
- ✓ Prefer integers in loops
- ✗ Don’t check for equality

```
https://www.securecoding.cert.org/confluence/
display/java/NUM09-J.+Do+not+use+floating-point+
variables+as+loop+counters
```

Do not construct BigDecimal objects from floating-point literals

- ✗ Floating-point numbers may be non-representable in base 10
- ✓ Start directly from BigDecimal if you need precision

```
https://www.securecoding.cert.org/confluence/
display/java/NUM10-J.+Do+not+construct+BigDecimal+
objects+from+floating-point+literals
```

Do not compare or inspect the string representation of floating-point values

- ✗ Floating-point numbers have different representations
- ✓ Compare BigDecimals
- ✓ Convert strings into BigDecimals

```
https://www.securecoding.cert.org/confluence/
display/java/NUM11-J.+Do+not+compare+or+inspect+
the+string+representation+of+floating-point+values
```

Ensure conversions of numeric types to narrower types do not result in lost or misinterpreted data

- ✗ Downcast is a source of mistakes
- ✓ Check values before downcasting

```
https://www.securecoding.cert.org/confluence/
display/java/NUM12-J.+Ensure+conversions+of+
numeric+types+to+narrower+types+do+not+result+in+
lost+or+misinterpreted+data
```

Avoid loss of precision when converting primitive integers to floating-point

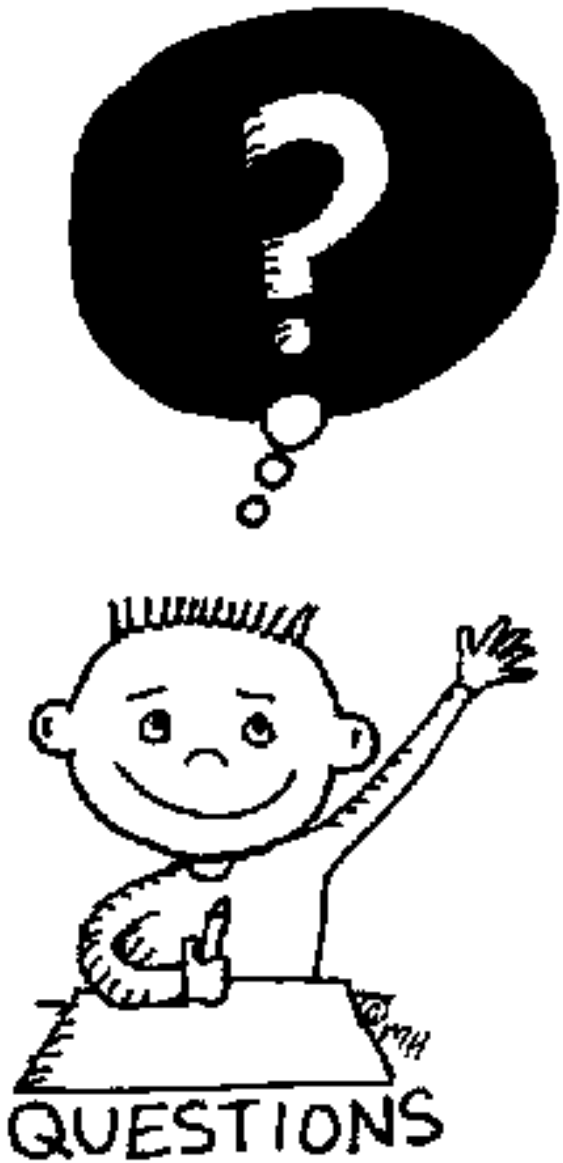
- ✗ Large integers may loss precision
- ✓ Check for these cases

```
https:
//www.securecoding.cert.org/confluence/display/
java/NUM13-J.+Avoid+loss+of+precision+when+
converting+primitive+integers+to+floating-point
```

Use shift operators correctly

- ✓ Use `>>>` for logical bit shift
- ✓ Use `>>` for (signed) arithmetic shift

```
https:
//www.securecoding.cert.org/confluence/display/
java/NUM14-J.+Use+shift+operators+correctly
```



END OF THE LECTURE