CMSC131

Conversions:
Widening and Narrowing
and Casting

Which of these seem valid?
(click all that seem valid)

0% 1. int i = 3.14;
0% 2. float f = 6;
0% 3. String s = 's';
0% 4. long j = 0F;
0% 5. String t = true;
Type Conflicts vs. Conversions

- There are two classifications of automatic conversion attempts; widening and narrowing.
  - Widening is valid in Java.
  - Narrowing is invalid in Java.

- The hierarchy for primitives in Java is:
  - byte → short → int → long → float → double

- For classes, we'll see how initialization is done soon…

Casting

- The widening of values can be done automatically (as in `float f = 6;`) but we can also explicitly cast values from one type to another in certain situations.
  ```
  int i = (int)(3.14);
  float f = (float)(1) / 5;
  ```

- In the above examples:
  - The value 3.14 is forced into being an integer so no automatic conversion was needed.
  - The value 1 was forced into being a floating point value which lead to float-division being called, which lead to 5 being widened to a floating point value.
Which are valid?

0% 1. A only
0% 2. B only
0% 3. A and B
0% 4. neither A nor B

(A) int q = 7;
byte b = q;

(B) short x = 17;
int y = x;

Copyright © 2010-2011 : Evan Golub