

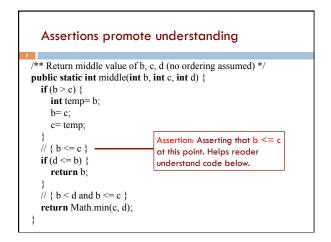
References to text and JavaSummary.pptx Local variable: variable declared in a method body B.10-B.11 slide 45 Inside-out rule, bottom-up/overriding rule C.15 slide 31-32 and consequences thereof slide 45 Use of this B.10 slide 23-24 and super C.15 slide 28, 33 Constructors in a subclass C.9-C.10 slide 24-29 First statement of a constructor body must be a call on another constructor —if not Java puts in super(); C.10 slide 29

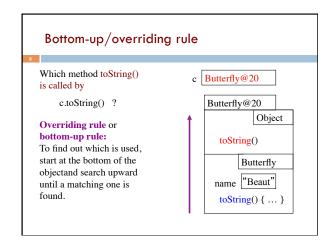
Homework 1. Visit course website, click on Resources and then on Code Style Guidelines. Study 4.2 Keep methods short 4.3 Use statement-comments ... 4.4 Use returns to simplify method structure 4.6 Declare local variables close to first use ...

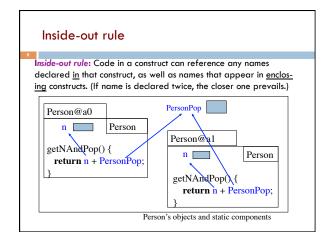
```
Local variables
                                                  middle(8, 6, 7)
/** Return middle value of b, c, d (no ordering assumed) */
public static int middle(int b, int c, int d) {
                                               Parameter: variable
  if (b > c) {
                                                  declared in () of
     int temp= b;
                          Local variable:
                                                   method header
     b=c;
                                variable
     c= temp;
                                              b 8 c 6 d 7
                             declared in
                           method body
  // \{ b \le c \}
                                                 temp ?
  if (d \le b) {
                             All parameters and local variables
     return b;
                             are created when a call is executed,
                             before the method body is executed.
  // \{ b < d \text{ and } b \le c \}
                             They are destroyed when method
  return Math.min(c, d);
                             body terminates.
```

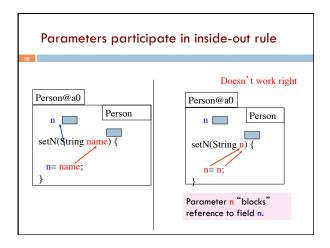
```
Scope of local variable
/** Return middle value of b, c, d (no ordering assumed) */
public static int middle(int b, int c, int d) {
  if (b > c) {
     int temp= b;
                             block
     b= c;
     c= temp;
  // \{ b \le c \}
                                 Scope of local variable (where it
  if (d \le b) {
                                can be used): from its declaration
     return b;
                                to the end of the block in which it
                                is declared.
  // \{ b < d \text{ and } b \le c \}
  return Math.min(c, d);
```

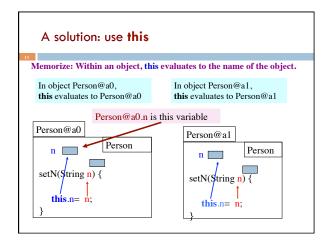
```
Principle about placement of declaration
/** Return middle value of b, c, d (no ordering assumed) */
public static int middle(int b, int c, int d) {
  int temp;
                             Not good! No need for reader to
  if (b > c) {
                             know about temp except when
    temp=b;
                             reading the then-part of the if-
     b=c;
                             statement
    c= temp;
  // \{ b \le c \}
  if (d \le b) {
    return b;
                             Principle: Declare a local variable
                             as close to its first use as possible.
  // \{ b < d \text{ and } b \le c \}
  return Math.min(c, d);
```

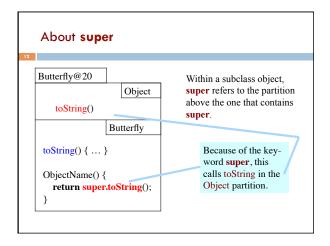












```
public class Time
private int hr; //hour of day, 0..23
private int min; // minute of hour, 0..59

/** Constructor: instance with h hours and m minutes */
public Time(int h, int m) { ...}

/** Constructor: instance with m minutes ... */
public Time(int m) {
    hr= m / 60;
    min= m % 60;
}

Want to change body
    to call first constructor
}
```

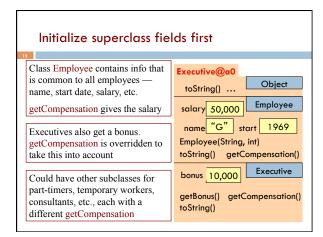
```
public class Time
private int hr; //hour of day, 0..23
private int min; // minute of hour, 0..59

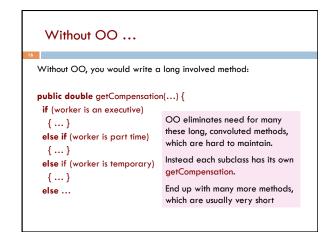
/** Constructor: instance with h hours and m minutes ... */
public Time(int h, int m) { ...}

/** Constructor: instance with m minutes ... */
public Time(int m) {
    this(m / 60, m % 60);
}

Use this (Instead of Time) to call another constructor in the class.

Must be first statement in constructor body!
```





Principle: initialize superclass fields first

/** Constructor: employee with name n, year hired d, salary s */
public Employee(String n, int d, double s) {
 name= n;
 start= d;
 salary= s;
}

Executive@a0
toString() ...
Salary 50,000 Employee
name "G" start 1969
Employee(String, int, double)

bonus 10,000 Executive
Executive(String, int, double)

Principle: initialize superclass fields first ** Constructor: employee with name n, year hired d, salary s */ public Employee(String n, int d, double s) /** Constructor: executive with name n, year hired d, salary of \$50,000, bonus b */ public Executive(String n, int d, Executive@a0 double s, double b) Employee salary Principle: In subclass constructor, name start fill in the superclass fields first Employee(String, int, double) How to do that if they are private? Executive Call constructor in superclass Executive(String, int, double)

