CS1110    Thursday, 16 February 2010

Congratulations!! You now know the basics of OO (object-orientation).

Discussion of Methods: Executing method calls. If-statements. The return statement in a function. Local variables.
For this and next lecture: Read chapter 2, but NOT 2.3.8!!!
Do the self-review exercises in 2.3.4

The last slide concerns local variables – variables declared within a method body. We don’t have time to discuss them. You are responsible for knowing about local variables. Read pp. 76-78 (sec. 2.3.7).

Take advantage: see videos of the 11:15 lecture for CS1110 on www.VideoNote.com. Log in with your Cornell netid

We write programs in order to do things. Methods are the key “doers”.

/** An instance keeps information about a book chapter */
public class Chapter {
    // class invariant: meanings of fields and constraints on them
    private int number; // the chapter number, in range 0..100
    private String title; // chapter title
    private Chapter prev; // instance for the previous chapter
    // (null if no previous chapter)
    ...
}

To execute the call x.setAmt(50);
1. Draw a frame for the call.
2. Assign the value of the argument to the parameter (in the frame).
3. Execute the method body. (Look for variables in the frame; if not there, look in the place given by the scope box.)
4. Erase the frame for the call.

/** Constructor: a chapter with title t, number n, and previous chapter null. */
public Chapter(String t, int n) {
    title = t;
    number = n;
    previous = null;
}

/** An instance keeps information about a book chapter */
public class Chapter {
    // class invariant: meanings of fields and constraints on them
    private int number; // the chapter number, in range 0..100
    private String title; // chapter title
    private Chapter prev; // instance for the previous chapter
    // (null if no previous chapter)
    ...
}

To execute the call cash= y.getAmt();
1. Draw frame for call.
2. Assign value of argument to parameters (in the frame).
3. Execute method body. (Look for variables in the frame; if not there, look in the place given by the scope box.)
4. Erase the frame for call; use value of return-statement expression as function-call value.

The frame (the box) for a method call

Remember: Every method is in a folder (object) or in a file-drawer.

But how is a method call executed?
How do parameters and arguments work?

Memorize: a parameter is a variable that is declared within the parentheses of a method header.

The scope box contains:
For an instance method, the name of the object in which it resides
For a static method, the name of the class in which it is defined.

For an instance method, the name of the object in which it resides
For a static method, the name of the class in which it is defined.
new Chapter("Intro", 1)
1. Draw a frame for the call.
2. Assign arg values to pars.
3. Execute the method body.
4. Erase the frame for the call.

Chapter(String t, int n) {
    String d;
    1: d = t;
    2: title = d;
    3: number = n;
    4: previous = null;
}

Syntax:
if (<boolean expression>)
<statement>
Execution:
if the <boolean expression> is true, then execute the <statement>.

Idiom: if statements and multiple return statements
/** = smallest of b, c, d */
public static int smallest(int b, int c, int d) {
    // The smallest is either c or d
    if (c <= d) {
        return c;
    }
    // The smallest is d
    return d;
}

/** = max of x and y */
public static int max(int x, int y) {
    // Swap x and y to put the max in x
    if (x < y) {
        int temp;
        temp = x;
        x = y;
        y = temp;
    }
    return x;
}

/* Put smaller of x, y in z */
if (x < y) {
    z = x;
} else {
    z = y;
}

// Swap x, y to put larger in y */
if (x > y) {
    int t;
    t = x;
    x = y;
    y = t;
}

Scope of local variable is the places where it can be used. The scope is the sequence of statements following it within the containing “block”.
/** = the max of x and y */
public static int max(int x, int y) {
    // Swap x and y to put the max in x
    if (x < y) {
        int temp;
        temp = x;
        x = y;
        y = temp;
    }
    return x;
}