Reading:
  1. Dale, Chapter 13 (Section 2, 3)

Objectives:
  1. Continue recursion discussion
  2. Quiz #2

Concepts:
  1. Recursion
1. Continue recursion discussion
   • Recursive steps
     o The recursion step executes while the original call to the method is still open (it has not finished executing)
     o The recursion step can result in many more recursive calls, as the method divides each new sub problem into two conceptual pieces.
   • Base cases
     o For the recursion to eventually terminate, each time the method calls itself with a slightly simpler version of the original problem, the sequence of smaller and smaller problems must converge on the base case. At that point, the method recognizes the best case, returns a result to the previous copy of the method, and a sequence of returns ensues up to line until the original method call eventually returns the final result to the caller.

2. Quiz #2
Objective:
1. Practice recursion

Student Activities:
1. The Fabonacci series: 0, 1, 1, 2, 3, 5, 8, 13, 21 … begins with 0 and 1 and has the property that each subsequent Fabonacci number is the sum of the previous two Fabonacci numbers. Write a program that asks the user for an integer value and then returns the Fabonacci value of that number. Show your output for: 1, 2, 3, 10, 20, 30
//Fibonacci Series: Recursive fibonacci method
import java.awt.*;
import java.applet.Applet;

public class FibonacciTest extends Applet
{
    Label numLabel, resultLabel;
    TextField num, result;
    public void init()
    {
        numLabel = new Label ( "Enter an integer and press enter" );
        num = new TextField (10);
        resultLabel = new Label ("Fibonacci value is ");
        result = new TextField (15);
        result.setEditable (false);

        add (numLabel);
        add (num);
        add (resultLabel);
        add (result);
    }
    public Boolean action (Event e, Object o)
    {
        long number, fibonacciVal;

        number = Long.parseLong (num.getText());
        showStatus ("Calculating …");
        fibonacciVal = Fibonacci (number);
        showStatus ("Done");
        result.setText (Long.toString (fibonacciVal));
        return true;
    }

    // Recursive definition of method Fibonacci
    long Fibonacci (long n)
    {
        if ( n == 0 || n == 1) // base case
            return n;
        else
            return Fibonacci (n-1) + Fibonacci (n-2);
    }
}
Expected output:

Enter an integer and press enter 0
Fibonacci value is 0
Done

Enter an integer and press enter 1
Fibonacci value is 1
Done

Enter an integer and press enter 0
Fibonacci value is 0
Done

Enter an integer and press enter 2
Fibonacci value is 1
Done

Enter an integer and press enter 3
Fibonacci value is 2
Done

Enter an integer and press enter 10
Fibonacci value is 55
Done

Enter an integer and press enter 20
Fibonacci value is 6765
Done

Enter an integer and press enter 30
Fibonacci value is 832040
Done
1) Fill in the blanks:
   a) A method that calls itself either directly or indirectly is a _______ method
   b) In Java it is possible to have various methods with the same name that each operate on different types and/or numbers of arguments. This is called method ______.
   c) the two components of recursion are: ________________________________ and ________________________________.

2) Find the error in each of the following program segments and explain how the error can be corrected:
   a) 
      ```java
      int sum (int n)
      {
        if (n == 0)
          return 0;
        else
          n + sum (n-1);
      }
      ```
      Error: The result of `n + sum(n-1)` is not returned; `sum` returns an improper result.
      Correction: Rewrite the statement in the `else` clause as:
      ```java
      return n + sum (n-1)
      ```
   b) 
      ```java
      void product ()
      {
        int a=6, b=5, a=4, result;
        result = a*b*c;
        System.out.println ("Result is “ + result);
        Return result;
      }
      ```
      Error: The method returns a value when it is not supposed to
      Correction: Eliminate the `return` statement

---

**Quiz #2 Solution**

1)a) Recursive
1)b) Overloading
1)c) one testing for a base case, one that expresses the problem as a recursive call
2)a)Error: The result of `n + sum(n-1)` is not returned; `sum` returns an improper result.
   Correction: Rewrite the statement in the `else` clause as:
   ```java
   return n + sum (n-1)
   ```
2)b) Error: The method returns a value when it is not supposed to
   Correction: Eliminate the `return` statement