

# For Wednesday

- Read chapter 8, sections 1-3
- UNIX handout due
- Recommended problems:
  - Checkpoints 8.1-8.8

# Exam 2

- Legible corrections due Wednesday, April 1
- Must be completely correct to get credit
- Complete corrections on separate paper
- Turn in graded exam with corrections
- Worth 20 points on exam 2 (up to a maximum score of 105 – the original exam max)
- May get help on this from me or each other (but don't just show answers, please)
- ADT definition must NOT be a duplicate of mine – must be your words
- Expect linked list and stack coverage on exam 3 and on the final (which is comprehensive)

# Program 6

- Any questions?

# Towers of Hanoi

- Three poles with disks of different size
- May never move more than one disk at once
- May never put a larger disk on a smaller one
- Task is to move a stack of disks from one pole to another (may use the third pole for temporary storage of disks)

# How Methods Work (in memory)

- Activation record
  - created (brand new) each time method is called
  - contains information needed by method
  - space for variables in the method
  - put into stack - LIFO
- Stack overflow
  - occurs when stack too full

# Recursive Methods

- A method that calls itself
- With each call to the method, the parameters to the method change
- An alternative to looping (though less efficient in most programming languages)
- Two characteristics
  - 1. method defined in terms of itself, with each invocation working on **smaller** versions of the problem
  - 2. task has a **terminal case** that is nonrecursive

# Structure of a Recursive method

```
returnType MethodName(/* args */)
{
    if (terminating_condition)
        // terminal case or base case
    else
        // reducing case (or recursive case)
        // always includes a call to MethodName
}
```

# Factorial

- What's the math definition for factorial?

# Try It

- Write a recursive method to compute an exponent. The method should take two integer parameters: the base and the power. The function should return a long.

# Greatest Common Divisor

- The greatest common divisor of two numbers can be computed as follows:
  - If the numbers are equal, their GCD is their value
  - If the first is greater, the GCD is the GCD of the second and the first minus the second
  - If the second is greater, the GCD is the GCD of the first and the second minus the first

# Fibonacci Numbers

- The Fibonacci sequence of numbers works as follows:

$$x_0 = 1$$

$$x_1 = 1$$

$$x_2 = x_1 + x_0 = 1 + 1 = 2$$

$$x_3 = x_2 + x_1 = 2 + 1 = 3$$

$$x_4 = x_3 + x_2 = 3 + 2 = 5$$

etc.

- Write a recursive Java method to compute the nth Fibonacci number

# Efficiency and Recursion

# Recursive Method Tracing

- ```
int funA(int num)
{
    int answer;
    if (num == 10)
        answer = 100;
    else
        answer = funA(num-1) + num;
    return answer;
}
```

# Example 2

- ```
int funcB(int num)
{
    int answer;
    if (num == 3)
        answer = 5;
    else
        answer = funcB(num+1) * 2;
    return answer;
}
```