Modeling Recursion

**Content**

**Advanced programming structures**

**Managing Complexity**

Recursion is an interesting but sometimes difficult concept. When dealing with more advanced ideas like this, it’s important to manage complexity in order to design and implement complex programs.

This activity helps you organize recursion in your program. It’s advised that you use these ideas, and consider these questions, as you create more programs that implement recursion.

You have 20 minutes to complete this modeling project. As a group, you should take the first 5 minutes of your time to decide what your sample recursive method will be. If you’re having trouble coming up with your own idea, you can implement factorial, power, sum, or Fibonacci series. During this brainstorming session, you should consider:

* How many recursive steps your method will need to execute fully? (Remember, you have to write your methods each time to demonstrate your model to your teacher and the class!)
* Does your method have a base case *AND* a recursive case? Can everyone in your group identify them? (You will be quizzed during class!)

Have several members of your group write the final version of your code into the boxes that will become the call stack. (You will need to explain the call stack to your teacher.) Cut out the cards, and practice modeling your code’s execution as a group. Troubleshoot your presentation; make sure you can answer the following questions:

* What is the base case in your code?
* What is the recursive case in your code?
* Show and describe the step that causes the method to advance to the next round of recursion.
* What causes your method to terminate?
* Where does Java return to after a method has terminated?
* What is your final output?