**Scope & Recursion**

* **Global variable** – defined outside of any enclosing function/class etc.
  + Comes into existence when declared and is deleted at the end of the program
* **Local variable** (including params) – defined within a function/method body
  + Comes into existence when the function is called, deleted when the function terminates
* **Member/instance variable** – sub-part of a large variable that is an instance of a struct/class
  + Comes into existence when the instance is created, delted when the instance is deleted
* **Scope** – the extent to which an indentifier is visible
  + End of {} block, end of procedure/loop/if body etc.
  + When a new scope is entered, an **activation record (AR)/stack frame** is created for it
  + AR contains storage for params, the return value, local variables, and the location from which the call was made from
  + When the scope is exited, its AR is destroyed; control returns to the calling function
* **Recursion** – “divide and conquer” breaking down a large problem to smaller problems
  + Base case(s) – can be solved easily and directly
  + Reduction operator – makes the data “smaller”
  + Composition operator – compose the smaller solutions into the full answer
  + E.g.

int factorial (int n) {

if (n <= 1) {

return 1; //**base case**

}

else {

return n \* factorial (n-1);

// **composition** **reduction**

}

}

* + E.g. Towers of Hanoi
    - 64 differently-sized disks stacked on a pole, increasing in size from top to bottom
    - The disks must be moved one at time from one pole to another (total 3 poles available)
    - No larger disk may ever sit on top of a smaller disk
    - How long does it take?