DATA STRUCTURES : Stack

* array
* postfix notation
* stack

Stacks are a data structure where the last object added is the first object out. This is opposite to queues. A queue would be used for a photocopier : the jobs are printed in the order that they are received.
Stacks are used for memory allocation, as well as for matching {}, for function calls (hence “stack overflow”). Stacks are used for undo / CTRL-Z in editors — where the most recent change is the one that is undone first, as well as for stacking plates in a cupboard.

Both stacks and queues can be coded using an arraylist.
If we add the new items at the end of the array list, a queue removes them from the beginning, while a stack removes them from the end.

We’re not going to be programming a stack application now. I was going have your program an RPN calculator (which uses a stack), but Java can’t intercept single keystrokes unless we’re using a GUI. We may do this later, or not.

Recursion and Stacks/Queues

Consider a web crawler program that makes a list of all webpages on some website.

- main:
- make an arraylist of pages visited
- get starting URL
  - ★ open the webpage
  - is this URL already on the list of pages? If so, return
  - add URL to the list of pages visited
  - read each line
  - if you come to a line that has an <a> tag get the URL and run the method ★
  - now continue reading the page
  - if you come to a line that has an <a> tag get the URL and run the method ★
  - ...

You can see that if you have webpages that link to webapges etc, you could get a stack overflow error as you keep calling method ★ from inside itself (which is what recursion is).

A better way of doing this is reading a whole page at once and adding all of the URLs to an arraylist of URLs that need to be visited. This would be a queue (or stack). You then process each URL in the list, and when the list is empty, you've crawled the whole website.
You never need recursion – calling the same method from inside itself – so you don't get stack overflow errors.