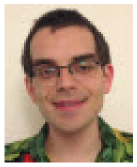
The Stack Data Structure in C and C++



By Alex Allain

The stack is a common data structure for representing things that need to maintained in a particular order. For instance, when a function calls another function, which in turn calls a third function, it's important that the third function return back to the second function rather than the first.

One way to think about this implementation is to think of functions as being stacked on top of each other; the last one added to the stack is the first one taken off. In this way, the data structure itself enforces the proper order of calls.

Conceptually, a stack is simple: a data structure that allows adding and removing elements in a particular order. Every time an element is added, it goes on the top of the stack; the only element that can be removed is the element that was at the top of the stack. Consequently, a stack is said to have "first in last out" behavior (or "last in, first out"). The first item added to a stack will be the last item removed from a stack.

So what's the big deal? Where do stacks come into play? As you've already seen, stacks are a useful way to organize our thoughts about how functions are called. In fact, the "call stack" is the term used for the list of functions either executing or waiting for other functions to return.

In a sense, stacks are part of the fundamental language of computer science. When you want to express an idea of the "first in last out" variety, it just makes sense to talk about it using the common terminology. Moreover, such operations show up an awful lot, from theoretical computer science tools such as a push-down automaton to AI, including implementations of depth-first search.

Stacks have some useful terminology associated with them:

- **Push** To add an element to the stack
- **Pop** To remove an element from the stock
- **Peek** To look at elements in the stack without removing them
- **LIFO** Refers to the last in, first out behavior of the stack
- **FILO** Equivalent to LIFO

Check out an implementation of a stack data structure using templates.

Next: Learn about queues

Back to algorithm and data structure tutorial index