

Re: recursive functions

Source: <http://coding.derkeiler.com/Archive/Perl/comp.lang.perl.misc/2004-08/0301.html>

From: Anno Siegel (anno4000_at_lublin.zrz.tu-berlin.de)

Date: 08/04/04

Date: 4 Aug 2004 19:18:05 GMT

Richard Gration <richard@zync.co.uk> wrote in comp.lang.perl.misc:

> *In article <cer0qa\$4um\$I@mamenchi.zrz.TU-Berlin.DE>, "Anno Siegel"*

> *<anno4000@lublin.zrz.tu-berlin.de> wrote:*

>> *Richard Gration <richard@zync.co.uk> wrote in comp.lang.perl.misc:*

>>> *The absolute classic example of a recursive algorithm is the factorial*

>>> *function. ...*

[...]

> *But you have to agree it illustrates the concept of recursion well. A*

The factorial example illustrates the *mechanics* of recursion. But so do summing up (instead of multiplying) the first integers, or counting the elements in a list, or a dozen tasks that can be done recursively, but are best done iteratively, in Perl and most other languages.

It does nothing to help develop judgment of when recursion is an appropriate solution, and where it's only (possibly expensive) claptrap. In everyday programming, the most frequent question about recursion is whether to use it at all. The factorial example teaches the wrong decision.

If you want a simple arithmetic example for a naturally recursive problem, use Euclid's algorithm.

```
sub euclid {  
    my ( $a, $b ) = @_;  
    $b ? euclid( $b, $a % $b ) : $a;  
}
```

> *computer is not the ideal tool to convert celsius to fahrenheit either,*

Why on earth not? Would you prefer a slide-rule? It isn't used to its full capacity, for sure, but that's something else.

> *yet I have written a few programs to do just that when learning a
> language. :-)*

comp.lang.perl.misc: Re: recursive functions

You weren't taught to do that recursively, I suppose :)

Anno