

## Re: Please help optimize (and standarize) this code...

*Source:* [http://coding.derkeiler.com/Archive/C\\_CPP/comp.lang.c/2005-03/1421.html](http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2005-03/1421.html)

---

*From:* Mark F. Haigh ([mfhaigh\\_at\\_sbcglobal.net](mailto:mfhaigh_at_sbcglobal.net))

*Date:* 03/12/05

Date: 11 Mar 2005 19:20:38 -0800

gtippery wrote:

> *Mark F. Haigh wrote:*

>

<large snip>

> *If you just wanted to show something similar, that's fine, and it was interesting. But the reason I didn't use an array of strings in the first place is that in the real program, the data's coming from an existing packed data structure in memory (returned by the OS), and would have to be converted to strings to use your sentinel and printing techniques; so as far as I can see I'd need the memcpy()'s anyway.*

And how was I supposed to know this? State this up front next time. I'd thought it was for some kind of class project (first name / 3 digit telephone extension), and that you had poor taste in data structures and compilers.

I'll take another look when I get some time. Disregard the previous post, and note:

1. Yes, you should cast `size_t` to `int` if it's going to be going through the default argument promotions. I shouldn't have posted so hastily right before bedtime (on some days, my only time for `c.l.c.`). Shame on me.
2. There are all kinds of dirty tricks and pre-canned "divide by x" code sequences for compiler writers. Look at "Hacker's Delight" by Henry S. Warren Jr. for more info on some interesting tricks (magic numbers, etc).
3. What's the biggest number of elements that your array can have? Check your platform docs. It may be more than `int` can handle, but less than or equal to what `size_t` can handle.

comp.lang.c: Re: Please help optimize (and standarize) this code...

4. Always put a return statement in main. It's necessary in C89, and good style in C99.

Mark F. Haigh  
mfhaigh@sbcglobal.net