

Re: Comment on trim string function please

## Re: Comment on trim string function please

---

*Source:* [http://coding.derkeiler.com/Archive/C\\_CPP/comp.lang.c/2008-07/msg01615.html](http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2008-07/msg01615.html)

---

- *From:* "Bill Reid" <[hormelfree@xxxxxxxxxxxxxxxxxx](mailto:hormelfree@xxxxxxxxxxxxxxxxxx)>
  - *Date:* Sun, 13 Jul 2008 15:20:14 GMT
- 

Ben Bacarisse <[ben.usenet@xxxxxxxxxx](mailto:ben.usenet@xxxxxxxxxx)> wrote in message  
[news:87ej5yk7d9.fsf@xxxxxxxxxxxxxxxx](mailto:news:87ej5yk7d9.fsf@xxxxxxxxxxxxxxxx)

"Bill Reid" <[hormelfree@xxxxxxxxxxxxxxxxxx](mailto:hormelfree@xxxxxxxxxxxxxxxxxx)> writes:

Ben Bacarisse <[ben.usenet@xxxxxxxxxx](mailto:ben.usenet@xxxxxxxxxx)> wrote in message  
[news:87iqvbjbeg.fsf@xxxxxxxxxxxxxxxx](mailto:news:87iqvbjbeg.fsf@xxxxxxxxxxxxxxxx)

"Bill Reid" <[hormelfree@xxxxxxxxxxxxxxxxxx](mailto:hormelfree@xxxxxxxxxxxxxxxxxx)> writes:

<snip>

In any event, does the following win the  
prize for most efficient  
implementation of the presumed  
requirements of the function?

I think you have a bug.

```
char *remove_beg_end_non_text(char *text)
{
    char *beg;
    size_t length;

    for(beg=text;*beg!='\0';beg++)
        if(!isspace(*beg)) break;
```

I'd guess

```
for (beg = text; isspace((unsigned char)*beg); beg++);
```

makes shorter code with some compilers.

Re: Comment on trim string function please

Yeah, but does it fly past the terminating null character?

Nope. Why would you think that?

Oh, you know, the fact that you aren't actually testing for the null character, you're just relying on isspace() to return 0 for it, which probably is correct in all cases, but just "looks dangerous"...

I think  
you may have out-obfuscated me...

I am happy to oblige. Those were your rules: you wanted minimal code; your own example suggests clarity does not matter (although I think my loop is clearer, but that is simply option).

Also, I've not been following why the cast is important here...

```
length=strlen(beg);  
  
while(length>0)  
if(!isspace(*(beg+(--length)))) break;  
  
*(beg+(++length))='\0';
```

If length never is never decremented (because it was zero after the initial space scan) then this writes outside the string. It always helps to walk through what your code does in boundary cases like "" and " ".

Actually, the "boundary case" for this is the size of array; as long as it is at least one character bigger than the "string" then this will always work. But you are correct if you replace the word "string" with the word "array fully-populated by the string" above.

There are actually only five possible test cases, which all should ASSUME a "array fully-populated by the string" (which I unfortunately didn't): spaces before text, spaces after the text, spaces before and after the text, just spaces, empty string (six if you count the

Re: Comment on trim string function please

stupidity

of passing NULL). Pass those five (or six) cases, the thing is perfect...

I am getting lost in all the words. I think your code is wrong (in two of the five cases you are considering). Are you saying it is correct?

No, it is not correct. You're right, it is a TRUE "bug".

Like many "off by one" bugs it may not produce undefined behaviour. For example passing `char test[2] = ""`; is fine because of the extra byte, but it is still a bug.

Of course. The way I allocate arrays for text on THIS system means you might NEVER encounter it, but it is still wrong...

```
return beg==text ? text :
memmove(text,beg,length+1);
}
```

Gotta admit, you couldn't reduce the cycles too much on that, could you? And it could even win a little bonus prize for obfuscatory conditions like `if(!isspace(*(beg+(--length))))...`

You might have obscure it even to yourself!

OK, I need to work on a few things to make it the perfect piece of obscure ruthlessly efficient code...

I'd want to be sure it is correct first. I would be the first admit I make mistakes but you have not persuaded me that I am wrong about this. I tried to be as clear about the bug as possible.

Don't worry about it, you're right. Also, I'm giving up on writing efficient obscure code for this. Aside from the cast "issues", here's something that is totally "straight-forward" but probably not the most "efficient" algorithm for all cases, or even any (it is also what

Re: Comment on trim string function please

Re: Comment on trim string function please

I actually use for small strings):

```
char *remove_beg_end_non_text(char *text) {
char *curr_char,*text_char=text;
size_t spaces=0;

for(curr_char=text;*curr_char!='\0';curr_char++)
if(!isspace(*curr_char)) break;

while(*curr_char!='\0') {

if(isspace(*curr_char)) spaces++;
else spaces=0;

*text_char++=*curr_char++;
}

*(text_char-spaces)='\0';

return text;
}
```

ASIDE FROM THE CAST "ISSUES", where's the bug in THAT?

---

William Ernest Reid

.