

Re: [C or C++] Is this legal? sizeof *p

Source: <http://coding.derkeiler.com/Archive/C/ CPP/alt.comp.lang.learn.c-cpp/2004-01/0139.html>

From: Jeff Schwab (jeffplus_at_comcast.net)

Date: 01/04/04

Date: Sun, 04 Jan 2004 01:38:42 -0500

Jack Klein wrote:

> *On Sat, 03 Jan 2004 17:17:30 -0500, Jeff Schwab <jeffplus@comcast.net>*

> *wrote in alt.comp.lang.learn.c-c++:*

>

>

>>*Jason wrote:*

>>

>>>{

>>>struct a_struct *p;

>>>

>>>p = malloc(sizeof *p);

>>>}

>>>

>>>*That should be legal shouldn't it? Despite the fact that p doesn't point to*

>>>*anything.*

>>>

>>>** For c++ p = static_cast<struct a_struct*> malloc(sizeof *p);*

>>>

>>>

>>

>>

>>*Nope. It usually works fine, but don't count on it. The pointer might*

>>*not hold a valid address, and the simple act of dereferencing it might*

>>*cause a bus error. Sorry.*

>

>

> *Jeff, you are completely wrong about this one.*

>

> *In C++, and in C except for one special case in C99, the sizeof*

> *operator is guaranteed not to evaluate its operand, merely use the*

> *operand to identify the type and hence the size.*

>

> *So this code is perfectly legal in both C and C++.*

>

> *The special case in C99 that I mentioned above does not apply here.*

> *That is for the a VLA (variable length array) which was added to C in*

> *1999, where the size of an automatic array is defined at run time. In*

> *that case if sizeof is applied to a VLA, it must obviously be*

alt.comp.lang.learn.c-c++: Re: [C or C++] Is this legal? sizeof *p

- > *evaluated at run time. But applying sizeof to a VLA is applying*
- > *sizeof to an array, not a pointer, and using the name of an array as*
- > *an operand of the sizeof operator is one of the few situations when*
- > *the array name is NOT silently converted to a pointer.*
- >

D'oh! Thank you for correcting me. Of course you are entirely right.