

Re: why does this work ?

Source: http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2003-10/1819.html

From: Barry Schwarz (schwarzb_at_deloz.net)

Date: 10/11/03

Date: 11 Oct 2003 17:17:28 GMT

On Sat, 11 Oct 2003 12:53:07 +0200, Sidney Cadot <sidney@jigsaw.nl> wrote:

>Joona I Palaste wrote:

>

>> <snip>

>> *There is no requirement for implementations to *have* a stack in the first place.*

>

>*This is the second time I see this posted over the last couple of days,*

>*and you're surely right. But it does beg the following question:*

>

>-----

>

>`#include <stdio.h>`

>

>`/* returns n! modulo 2^(number of bits in an unsigned long) */`

>`unsigned long f(unsigned long n)`

>`{`

>`return (n==0) ? 1 : f(n-1)*n;`

>`}`

>

>`int main(void)`

>`{`

>`unsigned long z;`

>`for(z=1;z!=0;z*=2)`

>`{`

>`printf("%lu %lu\n", z, f(z));`

>`fflush(stdout);`

>`}`

>`return 0;`

>`}`

>

>-----

>

>*As far as I can see, this is a perfectly valid C program that should*

>*reach the 'return 0' statement always, were it not for the fact that in*

comp.lang.c: Re: why does this work ?

It is valid in the theoretical sense but it executes in the real world.

*>all compilers I tried (one, actually) it terminates with a segmentation
>fault of some sort, due to limited stack size.*

>

*>Is this 'incorrect behavior' of all these compilers, or is there some
>wording in the Standard that covers for machines with a finite stack
>(much to my dismay, this covers all machines I have access to)?*

>

*>If so, is there a minimum depth of function calls that I can rely on to
>be executed properly? I would hate to rewr*