

## Re: Why segfault and no NULL match in for loop?

---

*Source:* [http://coding.derkeiler.com/Archive/C\\_CPP/comp.lang.c/2007-05/msg00192.html](http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2007-05/msg00192.html)

---

- *From:* jaysome <jaysome@xxxxxxxxxxxx>
  - *Date:* Thu, 03 May 2007 00:18:50 -0700
- 

On Wed, 02 May 2007 20:14:34 -0400, CBFalconer <cbfalconer@xxxxxxxx> wrote:

somebody wrote:

There are two files below named search.c and search.h. In the for loop in search.c, the for loop never exits, even if mystruct[i].field1 has no match. Instead of exiting the for loop it keeps going until it segfaults. This seems to be related to the strcmp with the NULL value. There are 2 comments below that indicate the segfaults. I guess the question is, when there is no match, how to I detect that and return without a segfault?

... snip to content of search.h ...

```
struct _mystruct mystruct[] =
{
  "AAA", "EEE", "R", "DF", 25,
  "BBB", "FFF", "R", "DF", 25,
  "CCC", "GGG", "R", "DF", 99,
  "DDD", "HHH", "R", "DF", 13,
};
```

NEVER define an actual object in a .h file. Doing so will cause it to be multiply defined, and result in link or run time errors. Add the word "extrn" to the definition, and delete the initialization. Put the above structure in ONLY one of the files in which you #include "search.h".

In addition to the correction noted by Keith ("extrn" should be "extern"), it should be noted that defining an actual object in a .h file does not necessarily mean that it will be multiply defined and result in a link or run time error.

## Re: Why segfault and no NULL match in for loop?

If the header file is included by a single translation unit, there will be no link or run time errors. You only run into trouble when multiple translation units include said header file and *\*think\** they are accessing the same data (something I have witnessed in practice).

Why would you ever want to define an actual object in a .h file? I can think of only one instance, which is if the .h file is shared between two or more programs (and included by a single translation unit in each, of course). I've used this technique, but only sparingly. Think of gcc and gcov using the same objects with file scope internal linkage, as an example (I'm not saying they do this, just that it is an acceptable option,