I was wondering if anyone would like to augment the information I will receive from a series of hopeful google searches with their own answer to this question:

When I declare a function which returns a value, but then I call that function in my code without assigning the returned value to anything... what happens to it? I don't get a compiler warning even with the extra, and so forth. Where in the green digital void does this value end up and why does the codeGod, usually so viciously fickle and cruel suddenly deem such an oversight perfectly acceptable?!

In example:

savageBurn *makeBurnMoreSavage(savageBurn *mildBurn, int desiredBurnSeverity);

But then I call the function without assigning the augmented burn, thusly:

savageBurn *lukeWarmInsult = newSavageBurn(char* iDoNotLikeYourInsults);
makeBurnMoreSavage(savageBurn *lukeWarmInsult, *9001);
displayBurn(savageBurn *lukeWarmInsult);

The original value is altered, and all is well. However, I wonder what becomes of the pointer that was supposed to be returned logically, but is ignored syntactically. Poor little guy... all alone out there... probably cold... scared.

There are compiler options that detect thrown-away return values. The conscientious programmer indicates that intent with a cast to void, as in:

savageBurn *lukeWarmInsult = newSavageBurn(char* iDoNotLikeYourInsults);
(void) makeBurnMoreSavage(savageBurn *lukeWarmInsult, *9001);
displayBurn(savageBurn *lukeWarmInsult);

Thank you for your response!
I have been using this method since you mentioned it in response to a fellow student's question in class.

If you have a moment to elaborate just a bit:

I wonder, in the case of no return set, and no void notation, does the value being 'returned' just get swept away with the other temporarily instantiated variables from the function when control returns to the call point?

I have found this a very tedious question to google with most explanations of what happens to the 'contents' of a called function after control returns not touching directly on this case.