Ryan Russell

Dr. Ahmed

CSCE.313.503

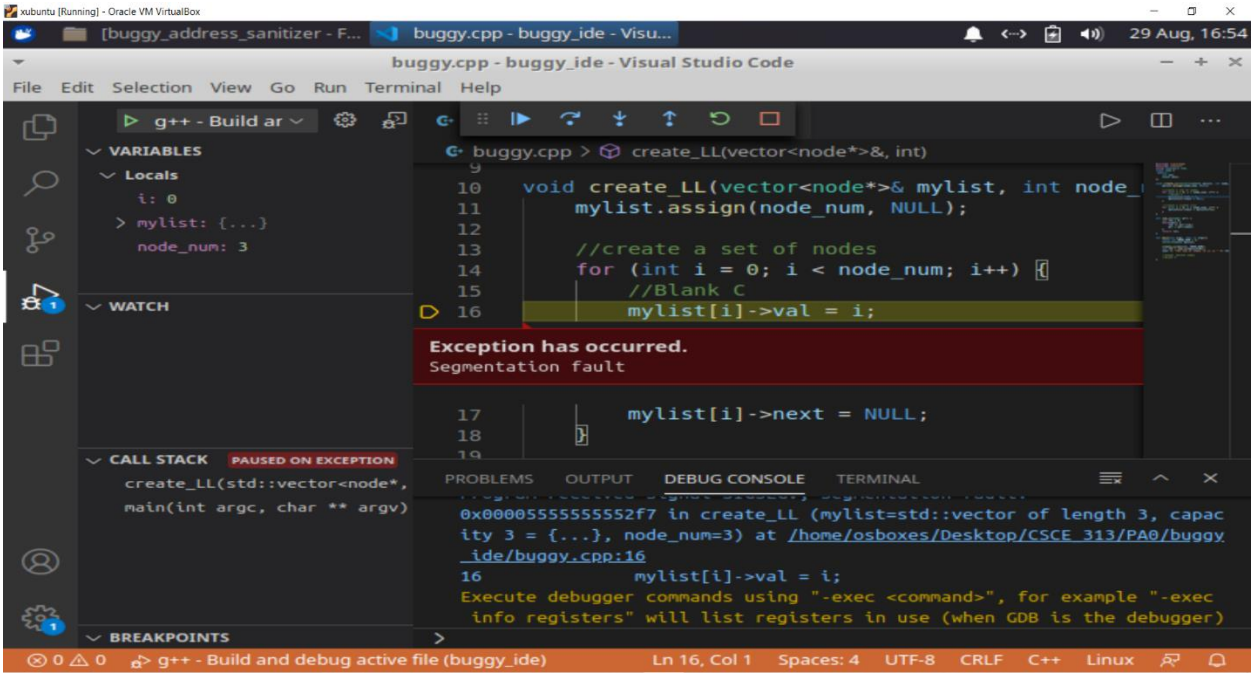
28 August 2020

Programming Assignment Zero Report

After fixing all the compilation errors in buggy.cpp, I began to debug the program with the gdb debugger. The first error that I caught with the gdb debugger was that there was a segmentation fault in line 16 (in the first for loop of the function). I used the backtrace command to see where did this error pop up else at in the program and found that it happened when the create_LL function was called. Then, I set up a breakpoint at line 15 where the for loop was created to see what the exact error was with line 16 by running the gdb again and then making steps into line 16. I realized that the error was because there were not any nodes being allocated to the memory. With this realization, I fixed this by including a line before line 16 where nodes were being allocated to the memory. The next error I received was that there was a segmentation fault in the function sum_LL (in the second line of the while loop in the function). After using the backtrace command, setting a breakpoint, and stepping to the line that had an error, I realized that the error was not in the sum_LL function. This led me to look back at the create_LL function and realize that the error stemmed from the second for loop of the function. The problem was that every node in the vector (including the last one) was pointing to the next; which meant that the last node was pointing to not NULL but to an empty element in the vector. This explained why the vector's size was 4 and the capacity was 6 when going through the debugger. I fixed this error by including an if/else statement in the for loop which would make

the last node's next node pointer variable point to NULL instead of an empty element in the vector. After that, the gdb debugger executed the program without any errors.

Starting over from scratch and fixing all the compilation errors again, I began to debug the program using the AddressSanitizer binary analysis tool. The first error that I received was that there was a SEGV error in the create_LL function. This meant that the program was dereferencing a null pointer which I fixed by allocating nodes in the create_LL function. The next error that I received was that there was a heap-buffer overflow in the create_LL function which meant that the program was trying to use a buffer to reach outside the allocation of the nodes. I fixed this by including an if/else statement that compensated for the last node pointing to NULL instead of an empty element in the vector. The final issue that came up from using the Address Sanitizer tool was that there was a direct leak of 16 bytes in 1 object and an indirect leak of 32 bytes in 2 objects all allocated in the create_LL function. This meant that the nodes allocated in the create_LL function were never properly deallocated in the program. I solved this error by creating a for loop at the end of the main that iterated through the vector and deallocated each node from memory.


xubuntu [Running] - Oracle VM VirtualBox
[buggy_address_sanitizer - F...
buggy.cpp - buggy_ide - Visu...
29 Aug, 16:54
buggy.cpp - buggy_ide - Visual Studio Code
File Edit Selection View Go Run Terminal Help
g++ - Build ar
VARIABLES
Locals
i: 0
mylist: {...}
node_num: 3
WATCH
CALL STACK
PAUSED ON EXCEPTION
create_LL(std::vector<node*,
main(int argc, char ** argv)
BREAKPOINTS
buggy.cpp > create_LL(vector<node*>&, int)
10 void create_LL(vector<node*>& mylist, int node_
11 mylist.assign(node_num, NULL);
12
13 //create a set of nodes
14 for (int i = 0; i < node_num; i++) {
15 //Blank C
16 mylist[i]->val = i;
Exception has occurred.
Segmentation fault
17 mylist[i]->next = NULL;
18 }
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
0x00005555555552f7 in create_LL (mylist=std::vector of length 3, capac
ity 3 = {...}, node_num=3) at /home/osboxes/Desktop/CSCE_313/PA0/buggy
_ide/buggy.cpp:16
16 mylist[i]->val = i;
Execute debugger commands using "-exec <command>", for example "-exec
info registers" will list registers in use (when GDB is the debugger)
0 0 g++ - Build and debug active file (buggy_ide)
Ln 16, Col 1 Spaces: 4 UTF-8 CRLF C++ Linux