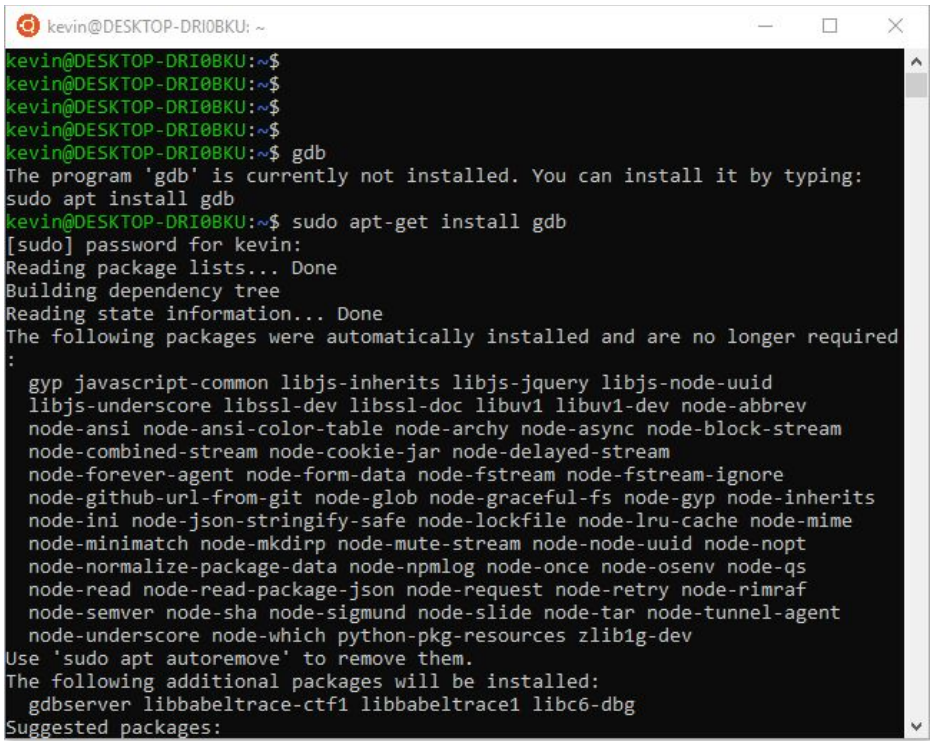
Kevin Tang
CSCE 313-511

Programming Assignment 0 Report

1. I used the linux ubuntu subsystem that windows 10 provides. I installed g++ and gdb onto the subsystem.



2. g++ buggy.cpp
3. Compile time error bugs:
    a. #include<vector> needs to be at the top of the program
    b. Add “using namespace std” so that the namespace is std by default
    c. Made all member variables public by add “public:” in the class definition
    d. Need to change “.” to “->” when accessing member variables of pointers
4.
```
kevin@DESKTOP-O1ULBEF:/mnt/c/Users/kevin/Google Drive/School/2019 Spring/CSCE 313/Programming Ass
ignments/PA0$ g++ buggy.cpp
kevin@DESKTOP-O1ULBEF:/mnt/c/Users/kevin/Google Drive/School/2019 Spring/CSCE 313/Programming Ass
ignments/PA0$ ls
PA0.pdf  a.out  buggy.cpp
kevin@DESKTOP-O1ULBEF:/mnt/c/Users/kevin/Google Drive/School/2019 Spring/CSCE 313/Programming Ass
ignments/PA0$ ./a.out
Segmentation fault (core dumped)
```
5. g++ -g buggy.cpp

6.
```
kevin@DESKTOP-O1ULBEF:/mnt/c/Users/kevin/Google Drive/School/2019 Spring/CSCE 313/Programming Ass
ignments/PA0$ g++ -g buggy.cpp
kevin@DESKTOP-O1ULBEF:/mnt/c/Users/kevin/Google Drive/School/2019 Spring/CSCE 313/Programming Ass
ignments/PA0$ gdb ./a.out
GNU gdb (Ubuntu 8.1-0ubuntu3) 8.1.0.20180409-git
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./a.out...done.
(gdb)
(gdb) ./a.out
Undefined command: "".  Try "help".
(gdb) run
Starting program: /mnt/c/Users/kevin/Google Drive/School/2019 Spring/CSCE 313/Programming Assignm
ents/PA0/a.out

Program received signal SIGSEGV, Segmentation fault.
0x0000000008000bb4 in create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3)
    at buggy.cpp:25
25                      mylist[i]->val = i;
(gdb) where
#0  0x0000000008000bb4 in create_LL (mylist=std::vector of length 3, capacity 3 = {...},
    node_num=3) at buggy.cpp:25
#1  0x0000000008000cbc in main (argc=1, argv=0x7fffffffee3a8) at buggy.cpp:48
(gdb) print(mylist[i])
$1 = (node *) 0x0
(gdb)
```

7.
```
(gdb) break 25
Breakpoint 1 at 0x8000b99: file buggy.cpp, line 25.
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /mnt/c/Users/kevin/Google Drive/School/2019 Spring/CSCE 313/Programming Assignm
ents/PA0/a.out

Breakpoint 1, create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3)
    at buggy.cpp:25
25                      mylist[i]->val = i;
(gdb) print(mylist[i])
$2 = (node *) 0x0
```

8. Fixed the segmentation error by assigning new node to the list and allocated memory
9. The segmentation fault is caused by accessing memory in the vector passed it's size. This can be fixed in the for loop by making it loop until myList.size - 1.
10. This can be done by creating a for loop and deleting every node memory that was allocated to the linked list.

Summary

The compile time issues were easily resolved because they were mostly syntax. The compiler gave a good description on what the exact problem was and why it was causing a compile time issue. Run time issues were more challenging and required gdb. For the first run time bug that caused a segmentation fault, running a backtrace and printing the list[i] object showed that it was a null pointer. This made it clear that there was no memory allocated to that object so it gave seg faults for trying to assign member variables of the null pointer. For the second run time bug, I suspected that the create linked list was accessing nodes in mylist past it's size. The for loop made this clear and after a backtrace, I noticed that it was setting the mylist[i].next (the last node) to mylist[i+1] which is past the list size. I changed the for loop so that it would terminate when i = node_num - 1. Overall these bugs were easy to find using the gdb functions.