

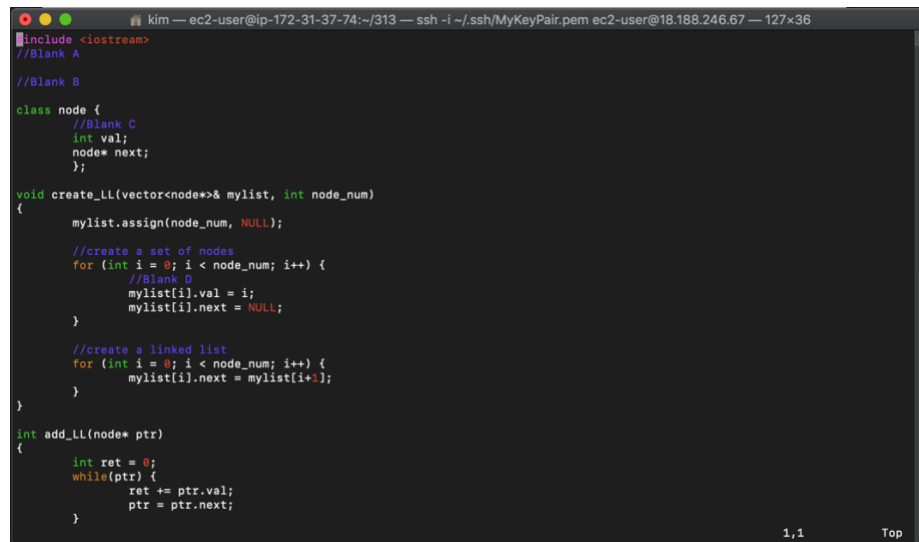
Kim Nguyen

CSCE 313-502

August 29, 2019

Report for PA0

1. System Setup



```
kim — ec2-user@ip-172-31-37-74:~/313 — ssh -i ~/.ssh/MyKeyPair.pem ec2-user@18.188.246.67 — 127x36
#include <iostream>
//Blank A

//Blank B

class node {
    //Blank C
    int val;
    node* next;
};

void create_LL(vector<node*>& mylist, int node_num)
{
    mylist.assign(node_num, NULL);

    //create a set of nodes
    for (int i = 0; i < node_num; i++) {
        //Blank D
        mylist[i].val = i;
        mylist[i].next = NULL;
    }

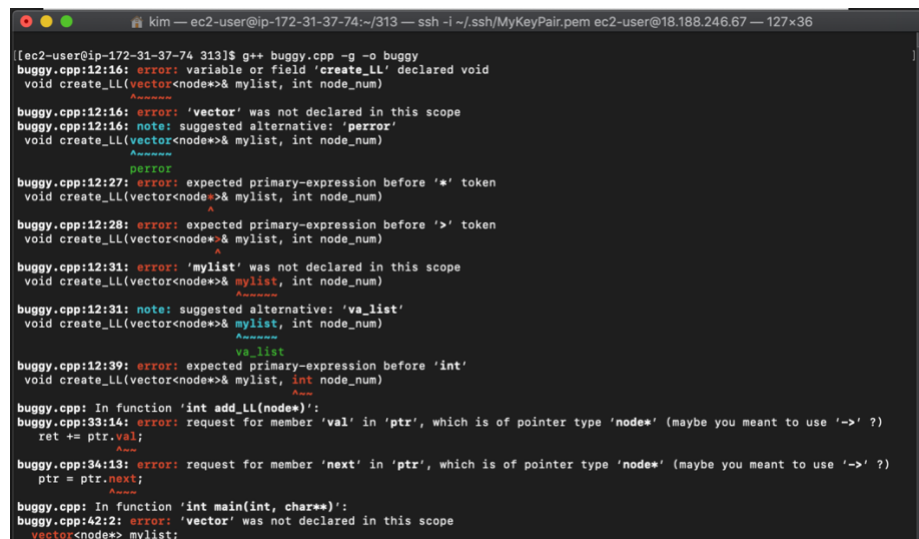
    //create a linked list
    for (int i = 0; i < node_num; i++) {
        mylist[i].next = mylist[i+1];
    }
}

int add_LL(node* ptr)
{
    int ret = 0;
    while(ptr) {
        ret += ptr.val;
        ptr = ptr.next;
    }
}
```

- I used AWS EC2 to set up a Linux virtual machine.

2. C++ Compilation

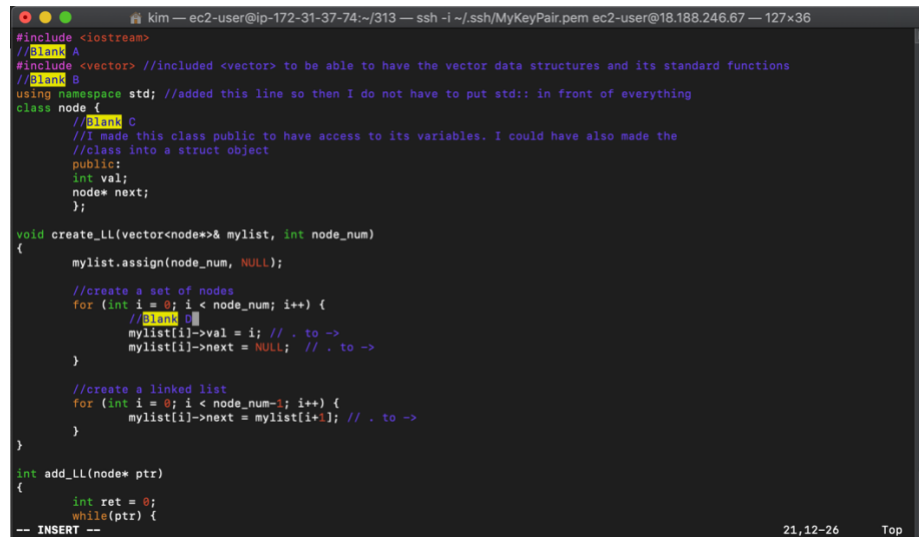
- For the command on how to compile the buggy.cpp file, I used g++ and -g for gdb and -o to make a Unix executable file. When I tried to compile



```
kim — ec2-user@ip-172-31-37-74:~/313 — ssh -i ~/.ssh/MyKeyPair.pem ec2-user@18.188.246.67 — 127x36
[ec2-user@ip-172-31-37-74 313]$ g++ buggy.cpp -g -o buggy
buggy.cpp:12:16: error: variable or field 'create_LL' declared void
void create_LL(vector<node*>& mylist, int node_num)
               ^~~~~~
buggy.cpp:12:16: error: 'vector' was not declared in this scope
buggy.cpp:12:16: note: suggested alternative: 'perror'
void create_LL(vector<node*>& mylist, int node_num)
               ^~~~~~
buggy.cpp:12:27: error: expected primary-expression before '*' token
void create_LL(vector<node*>& mylist, int node_num)
               ^
buggy.cpp:12:28: error: expected primary-expression before '>' token
void create_LL(vector<node*>& mylist, int node_num)
               ^
buggy.cpp:12:31: error: 'mylist' was not declared in this scope
void create_LL(vector<node*>& mylist, int node_num)
               ^~~~~~
buggy.cpp:12:31: note: suggested alternative: 'va_list'
void create_LL(vector<node*>& mylist, int node_num)
               ^~~~~~
buggy.cpp:12:39: error: expected primary-expression before 'int'
void create_LL(vector<node*>& mylist, int node_num)
               ^~~~~~
buggy.cpp: In function 'int add_LL(node*)':
buggy.cpp:33:14: error: request for member 'val' in 'ptr', which is of pointer type 'node*' (maybe you meant to use '->' ?)
    ret += ptr.val;
               ^
buggy.cpp:34:13: error: request for member 'next' in 'ptr', which is of pointer type 'node*' (maybe you meant to use '->' ?)
    ptr = ptr.next;
           ^
buggy.cpp: In function 'int main(int, char*)':
buggy.cpp:42:12: error: 'vector' was not declared in this scope
    vector<node*> mylist;
```

3. Compile Time Error Debugging

- I have made notes of my changes in the code's comments.



```
#include <iostream>
//Blank A
#include <vector> //included <vector> to be able to have the vector data structures and its standard functions
//Blank B
using namespace std; //added this line so then I do not have to put std:: in front of everything
class node {
//Blank C
//I made this class public to have access to its variables. I could have also made the
//class into a struct object
public:
    int val;
    node* next;
};

void create_LL(vector<node*> &mylist, int node_num)
{
    mylist.assign(node_num, NULL);

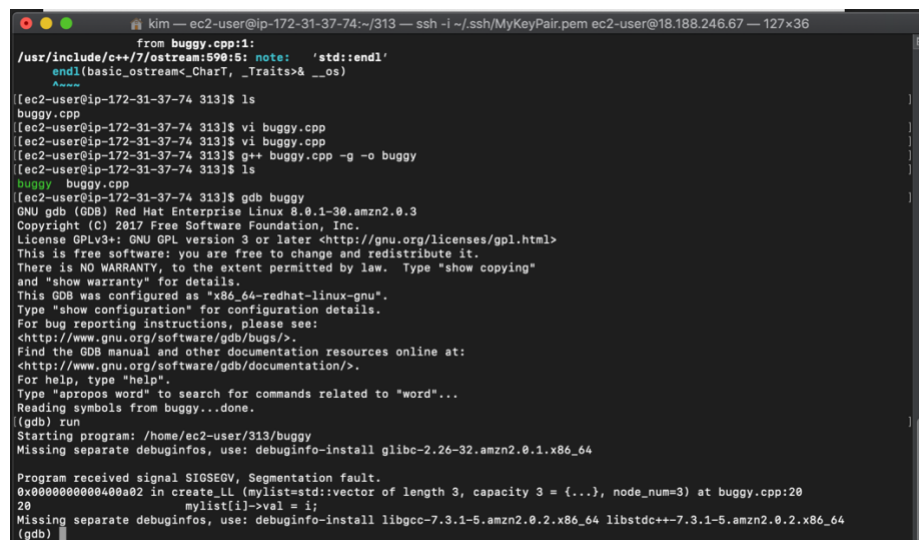
    //create a set of nodes
    for (int i = 0; i < node_num; i++) {
        //Blank D
        mylist[i]->val = i; // . to ->
        mylist[i]->next = NULL; // . to ->
    }

    //create a linked list
    for (int i = 0; i < node_num-1; i++) {
        mylist[i]->next = mylist[i+1]; // . to ->
    }
}

int add_LL(node* ptr)
{
    int ret = 0;
    while(ptr) {
        -- INSERT --
    }
}
```

4. C++ Program Execution

- I encountered a segmentation fault at line 20 and then the program stopped running.



```
from buggy.cpp:1:
/usr/include/c++/7/ostream:590:5: note: 'std::endl'
    endl(basic_ostream<_CharT, _Traits>& __os)
    ^~~~~
[ec2-user@ip-172-31-37-74 313]$ ls
buggy.cpp
[ec2-user@ip-172-31-37-74 313]$ vi buggy.cpp
[ec2-user@ip-172-31-37-74 313]$ g++ -g -o buggy
[ec2-user@ip-172-31-37-74 313]$ ls
buggy  buggy.cpp
[ec2-user@ip-172-31-37-74 313]$ gdb buggy
GNU gdb (GDB) Red Hat Enterprise Linux 8.0.1-30.amzn2.0.3
Copyright (C) 2017 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-redhat-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 buggy...done.
(gdb) run
Starting program: /home/ec2-user/313/buggy
Missing separate debuginfos, use: debuginfo-install glibc-2.26-32.amzn2.0.1.x86_64

Program received signal SIGSEGV, Segmentation fault.
0x0000000000000002 in create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3) at buggy.cpp:20
20      mylist[i]->val = i;
Missing separate debuginfos, use: debuginfo-install libgcc-7.3.1-5.amzn2.0.2.x86_64 libstdc++-7.3.1-5.amzn2.0.2.x86_64
(gdb)
```

5. C++ Compilation in Debug Mode

- The command I used was **b** using gdb in order to set a breakpoint and inserted an integer to signify which line of code I wanted to set a breakpoint in.
- To start running the program I used the **run** command and the **step** command whenever I am jumping through breakpoints.

```
kim — ec2-user@ip-172-31-37-74:~/313 — ssh -i ~/.ssh/MyKeyPair.pem ec2-user@18.188.246.67 — 127x36
GNU gdb (GDB) Red Hat Enterprise Linux 8.0.1-30.amzn2.0.3
Copyright (C) 2017 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-redhat-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 buggy...done.
(gdb) run
Starting program: /home/ec2-user/313/buggy
Missing separate debuginfos, use: debuginfo-install glibc-2.26-32.amzn2.0.1.x86_64

Program received signal SIGSEGV, Segmentation fault.
0x0000000000400a02 in create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3) at buggy.cpp:20
20      mylist[i]-->val = i;
Missing separate debuginfos, use: debuginfo-install libgcc-7.3.1-5.amzn2.0.2.x86_64 libstdc++-7.3.1-5.amzn2.0.2.x86_64
(gdb) b 19
Breakpoint 1 at 0x4009e7: file buggy.cpp, line 19.
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/ec2-user/313/buggy

Breakpoint 1, create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3) at buggy.cpp:20
20      mylist[i]-->val = i;
(gdb) backtrace
#0 create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3) at buggy.cpp:20
#1 0x0000000000400ae7 in main (argc=1, argv=0x7ffffffe498) at buggy.cpp:45
(gdb) █
```

6. GDB Start/Run/Backtrace

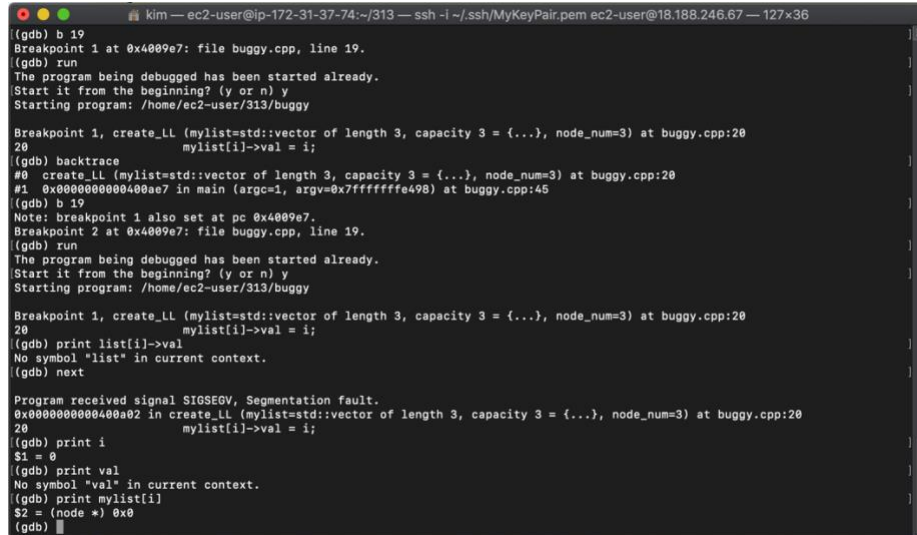
```
kim — ec2-user@ip-172-31-37-74:~/313 — ssh -i ~/.ssh/MyKeyPair.pem ec2-user@18.188.246.67 — 127x36
GNU gdb (GDB) Red Hat Enterprise Linux 8.0.1-30.amzn2.0.3
Copyright (C) 2017 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free 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-redhat-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 buggy...done.
(gdb) run
Starting program: /home/ec2-user/313/buggy
Missing separate debuginfos, use: debuginfo-install glibc-2.26-32.amzn2.0.1.x86_64

Program received signal SIGSEGV, Segmentation fault.
0x0000000000400a02 in create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3) at buggy.cpp:20
20      mylist[i]-->val = i;
Missing separate debuginfos, use: debuginfo-install libgcc-7.3.1-5.amzn2.0.2.x86_64 libstdc++-7.3.1-5.amzn2.0.2.x86_64
(gdb) b 19
Breakpoint 1 at 0x4009e7: file buggy.cpp, line 19.
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/ec2-user/313/buggy

Breakpoint 1, create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3) at buggy.cpp:20
20      mylist[i]-->val = i;
(gdb) backtrace
#0 create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3) at buggy.cpp:20
#1 0x0000000000400ae7 in main (argc=1, argv=0x7ffffffe498) at buggy.cpp:45
(gdb) █
```

7. GDB Breakpoint/Print

- I added a breakpoint to line 19 and used the **print** command in order to print the value of **i** and **mylist[i]**.



```
(gdb) b 19
Breakpoint 1 at 0x4009e7: file buggy.cpp, line 19.
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/ec2-user/313/buggy

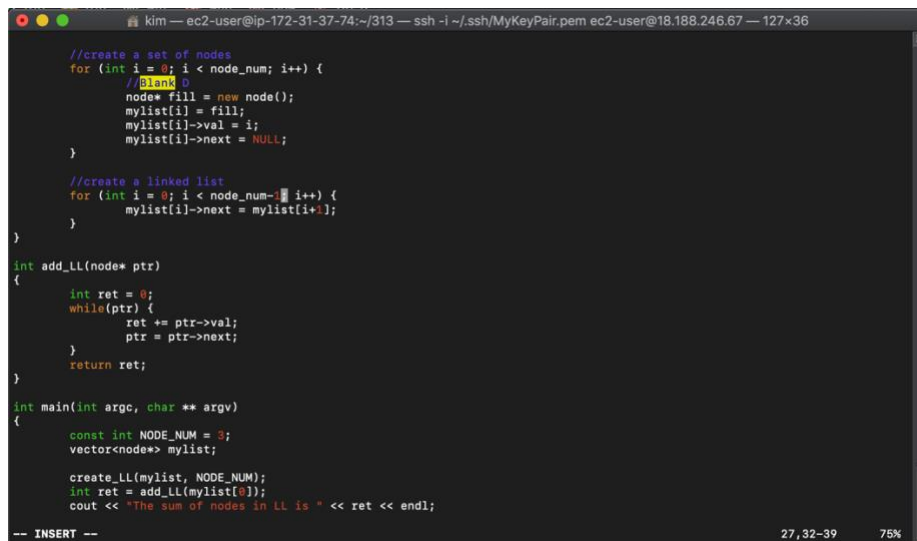
Breakpoint 1, create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3) at buggy.cpp:20
20      mylist[i]-->val = i;
(gdb) backtrace
#0  create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3) at buggy.cpp:20
#1  0x0000000000400ae7 in main (argc=1, argv=0x7fffffff498) at buggy.cpp:45
(gdb) b 19
Note: breakpoint 1 also set at pc 0x4009e7.
Breakpoint 2 at 0x4009e7: file buggy.cpp, line 19.
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/ec2-user/313/buggy

Breakpoint 1, create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3) at buggy.cpp:20
20      mylist[i]-->val = i;
(gdb) print list[i]-->val
No symbol "list" in current context.
(gdb) next

Program received signal SIGSEGV, Segmentation fault.
0x0000000000400ae7 in create_LL (mylist=std::vector of length 3, capacity 3 = {...}, node_num=3) at buggy.cpp:20
20      mylist[i]-->val = i;
(gdb) print i
$1 = 0
(gdb) print val
No symbol "val" in current context.
(gdb) print mylist[i]
$2 = (node *) 0x0
(gdb)
```

8. C++ Runtime Error Fix (Null-Pointer)

- Under **//Blank D** I made a new variable, **fill**, that is a dynamically empty node. I used this to initialize the current element of the vector in order to prevent the error that was occurring. What was happening was that there was nothing inside the vector and the program would error, so initializing the vector element prevented this.



```
//create a set of nodes
for (int i = 0; i < node_num; i++) {
    //Blank D
    node* fill = new node();
    mylist[i] = fill;
    mylist[i]-->val = i;
    mylist[i]-->next = NULL;
}

//create a linked list
for (int i = 0; i < node_num; i++) {
    mylist[i]-->next = mylist[i+1];
}

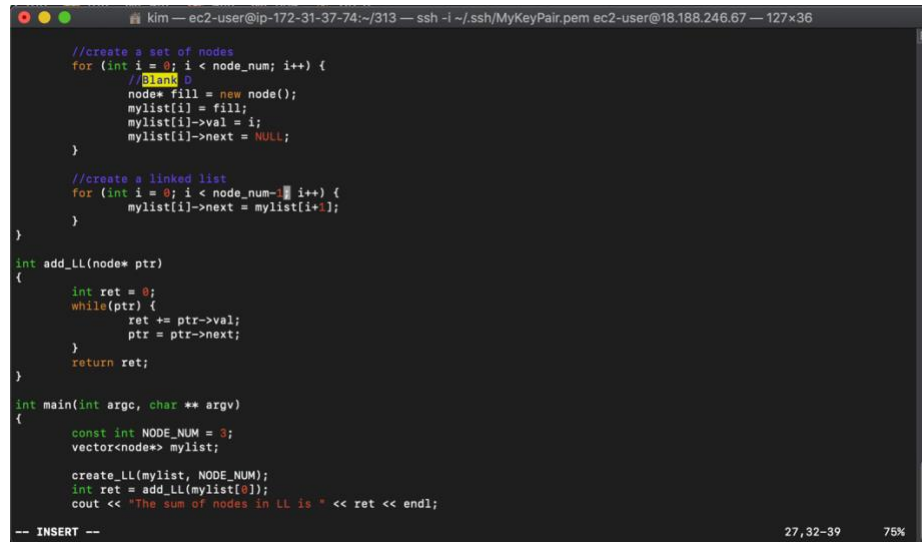
int add_LL(node* ptr)
{
    int ret = 0;
    while(ptr) {
        ret += ptr-->val;
        ptr = ptr-->next;
    }
    return ret;
}

int main(int argc, char ** argv)
{
    const int NODE_NUM = 3;
    vector<node*> mylist;

    create_LL(mylist, NODE_NUM);
    int ret = add_LL(mylist[0]);
    cout << "The sum of nodes in LL is " << ret << endl;
}
```

9. C++ Runtime Error Fix (non-NULL garbage value Pointer)

- In the second for loop of the **create_LL** function, the error occurred because of the **ptr->val** in **add_LL**. To stop this error from happening, I subtracted one from **node_num** in **create_LL**'s second for loop in order to prevent the program from going out of bounds.



```
kim — ec2-user@ip-172-31-37-74:~/313 — ssh -i ~/.ssh/MyKeyPair.pem ec2-user@18.188.246.67 — 127x36

//create a set of nodes
for (int i = 0; i < node_num; i++) {
    //Blank D
    node* fill = new node();
    mylist[i] = fill;
    mylist[i]->val = i;
    mylist[i]->next = NULL;
}

//create a linked list
for (int i = 0; i < node_num-1; i++) {
    mylist[i]->next = mylist[i+1];
}

int add_LL(node* ptr)
{
    int ret = 0;
    while(ptr) {
        ret += ptr->val;
        ptr = ptr->next;
    }
    return ret;
}

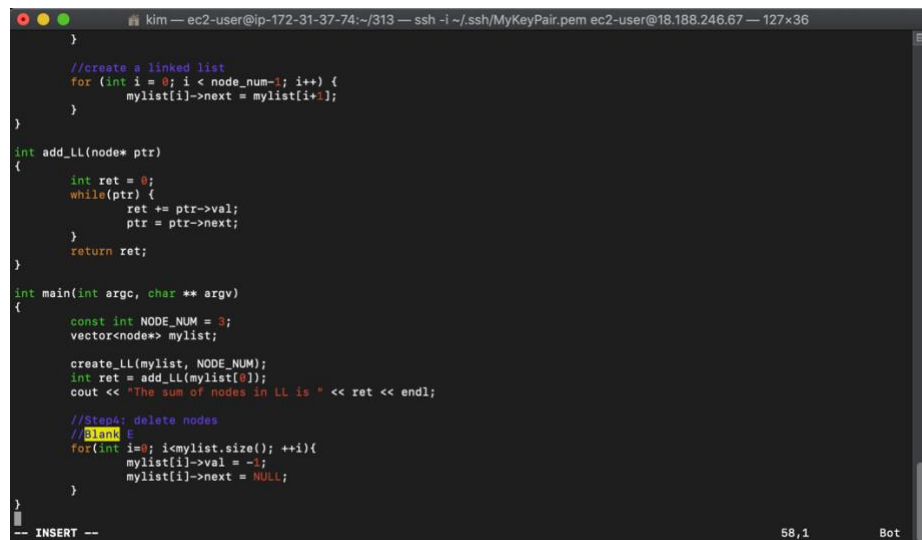
int main(int argc, char ** argv)
{
    const int NODE_NUM = 3;
    vector<node*> mylist;

    create_LL(mylist, NODE_NUM);
    int ret = add_LL(mylist[0]);
    cout << "The sum of nodes in LL is " << ret << endl;
}

-- INSERT --
```

10. Dynamically Allocated Memory Deletion

- Under **//Blank E** I coded a for loop that will make all of the **val**'s -1 and the next to be **NULL** to prevent a memory leak from happening.



```
kim — ec2-user@ip-172-31-37-74:~/313 — ssh -i ~/.ssh/MyKeyPair.pem ec2-user@18.188.246.67 — 127x36

//create a linked list
for (int i = 0; i < node_num-1; i++) {
    mylist[i]->next = mylist[i+1];
}

int add_LL(node* ptr)
{
    int ret = 0;
    while(ptr) {
        ret += ptr->val;
        ptr = ptr->next;
    }
    return ret;
}

int main(int argc, char ** argv)
{
    const int NODE_NUM = 3;
    vector<node*> mylist;

    create_LL(mylist, NODE_NUM);
    int ret = add_LL(mylist[0]);
    cout << "The sum of nodes in LL is " << ret << endl;

    //Stand: delete nodes
    //Blank E
    for(int i=0; i<mylist.size(); ++i){
        mylist[i]->val = -1;
        mylist[i]->next = NULL;
    }
}

-- INSERT --
```

In the end the changes I made to the code allowed it to start working. I have tested this by changing the value of `NODE_NUM` in the main function from 3 to several different values. I used the triangular sum formula to ensure that the program is computing the correct number every time.

```

kim — ec2-user@ip-172-31-37-74:~/313 — ssh -i ~/.ssh/MyKeyPair.pem ec2-user@18.188.246.67 — 127×36
(gdb) print mylist[i]
$2 = (node *) 0x0
(gdb) q
A debugging session is active.

Inferior 1 [process 18173] will be killed.

Quit anyway? (y or n) y
[ec2-user@ip-172-31-37-74 313]$ vi buggy.cpp
[ec2-user@ip-172-31-37-74 313]$ ls
buggy  buggy.cpp
[ec2-user@ip-172-31-37-74 313]$ rm buggy
[ec2-user@ip-172-31-37-74 313]$ g++ buggy.cpp -g -o buggy
[ec2-user@ip-172-31-37-74 313]$ gdb buggy
GNU gdb (GDB) Red Hat Enterprise Linux 8.0.1-30.amzn2.0.3
Copyright (C) 2017 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-redhat-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 buggy...done.
(gdb) run
Starting program: /home/ec2-user/313/buggy
Missing separate debuginfos, use: debuginfo-install glibc-2.26-32.amzn2.0.1.x86_64
The sum of nodes in LL is 3
[Inferior 1 (process 18203) exited normally]
Missing separate debuginfos, use: debuginfo-install libgcc-7.3.1-5.amzn2.0.2.x86_64 libstdc++-7.3.1-5.amzn2.0.2.x86_64
(gdb)

```

This was when `NODE_NUM = 3`.

```

kim — ec2-user@ip-172-31-37-74:~/313 — ssh -i ~/.ssh/MyKeyPair.pem ec2-user@18.188.246.67 — 127×36
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from buggy...done.
(gdb) run
Starting program: /home/ec2-user/313/buggy
Missing separate debuginfos, use: debuginfo-install glibc-2.26-32.amzn2.0.1.x86_64
The sum of nodes in LL is 3
[Inferior 1 (process 19848) exited normally]
Missing separate debuginfos, use: debuginfo-install libgcc-7.3.1-5.amzn2.0.2.x86_64 libstdc++-7.3.1-5.amzn2.0.2.x86_64
(gdb) q
[ec2-user@ip-172-31-37-74 313]$ vi buggy.cpp
[ec2-user@ip-172-31-37-74 313]$ rm buggy
[ec2-user@ip-172-31-37-74 313]$ g++ buggy.cpp -g -o buggy
[ec2-user@ip-172-31-37-74 313]$ gdb buggy
GNU gdb (GDB) Red Hat Enterprise Linux 8.0.1-30.amzn2.0.3
Copyright (C) 2017 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-redhat-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 buggy...done.
(gdb) run
Starting program: /home/ec2-user/313/buggy
Missing separate debuginfos, use: debuginfo-install glibc-2.26-32.amzn2.0.1.x86_64
The sum of nodes in LL is 44850
[Inferior 1 (process 19861) exited normally]
Missing separate debuginfos, use: debuginfo-install libgcc-7.3.1-5.amzn2.0.2.x86_64 libstdc++-7.3.1-5.amzn2.0.2.x86_64
(gdb)

```

This was when `NODE_NUM = 300`.

~~Proof: $300 \cdot (300 + 1) / 2 = 44850$~~

Proof: $299 \cdot (299 - 1) / 2 = 44850$ (We do `NODE_NUM - 1` as the variable because it starts as $0 + 1 + \dots$ instead of $1 + 2 + \dots$)