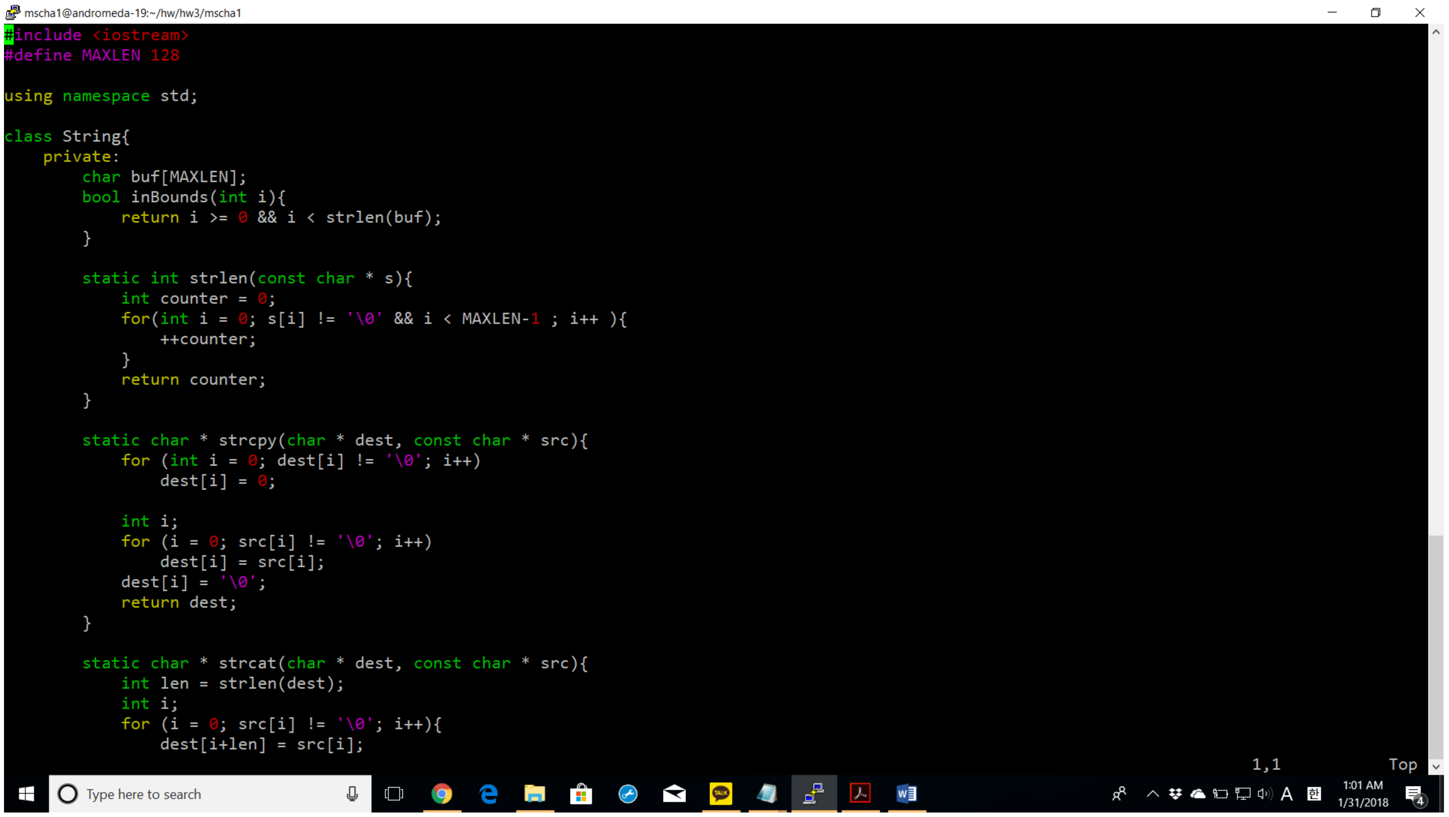
Name: Min Sung Cha Student Id: 85408485

Analysis of a few functions and any embedded functions

String.h

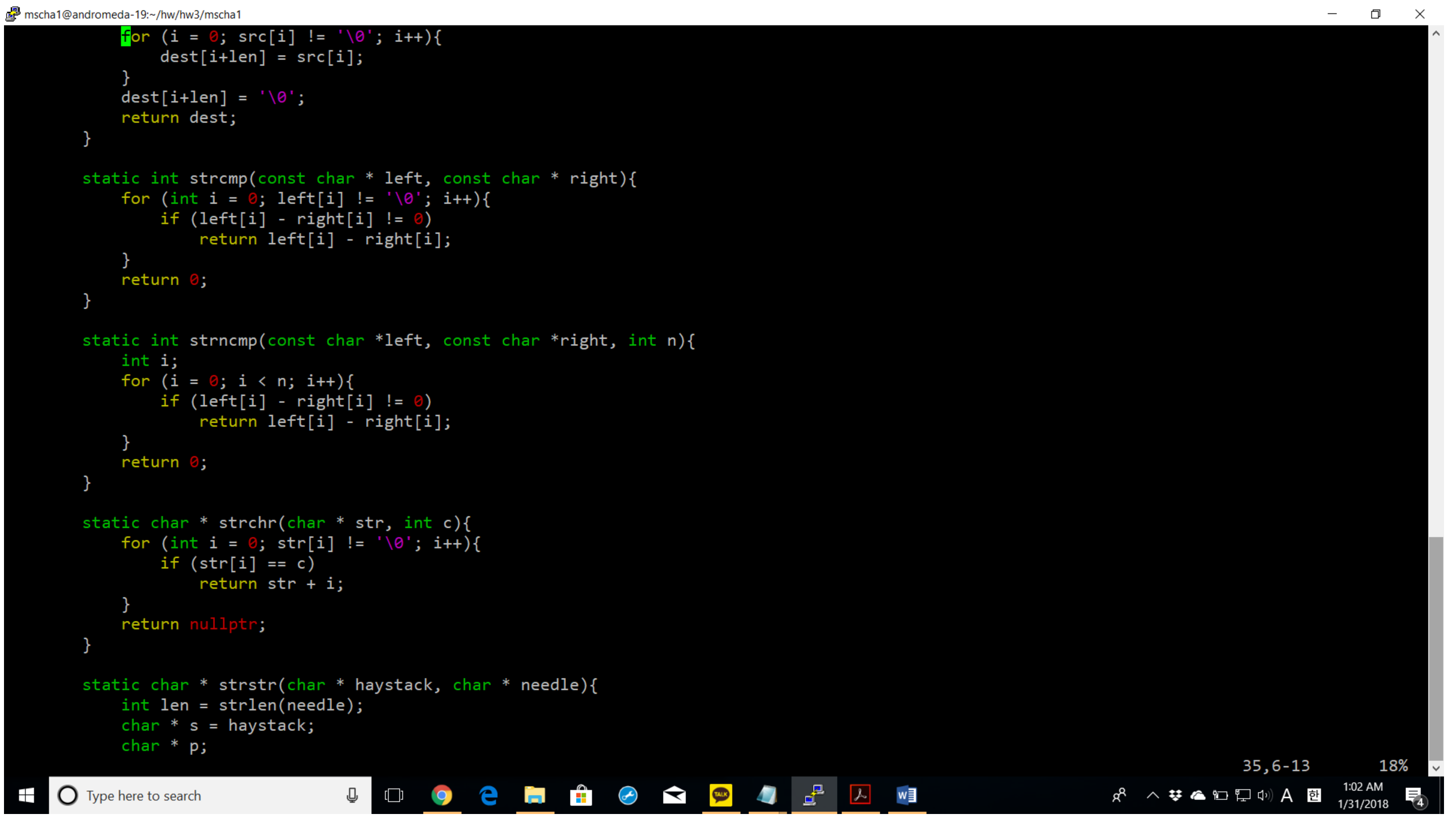


mscha1@andromeda-19:~/hw/hw3/mscha1

```
#include <iostream>
#define MAXLEN 128

using namespace std;

class String{
    private:
        char buf[MAXLEN];
        bool inBounds(int i){
            return i >= 0 && i < strlen(buf);
        }

        static int strlen(const char * s){
            int counter = 0;
            for(int i = 0; s[i] != '\0' && i < MAXLEN-1 ; i++ ){
                ++counter;
            }
            return counter;
        }

        static char * strcpy(char * dest, const char * src){
            for (int i = 0; dest[i] != '\0'; i++)
                dest[i] = 0;

            int i;
            for (i = 0; src[i] != '\0'; i++)
                dest[i] = src[i];
            dest[i] = '\0';
            return dest;
        }

        static char * strcat(char * dest, const char * src){
            int len = strlen(dest);
            int i;
            for (i = 0; src[i] != '\0'; i++){
                dest[i+len] = src[i];
                                                                                           1,1           Top
```

strlen: calculates the length of an array until the null character.

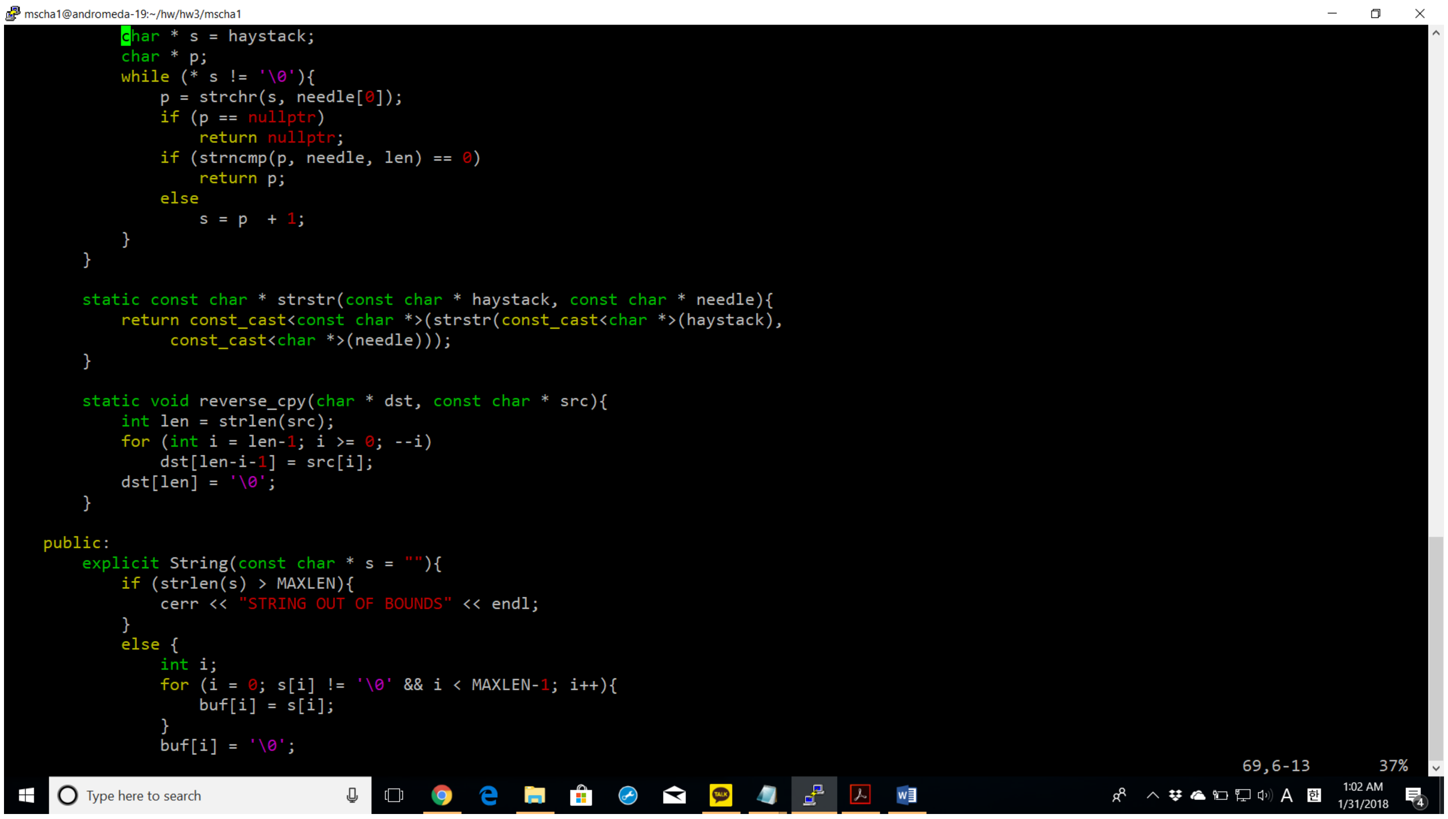
strcpy: copies the string from src to destination.



mscha1@andromeda-19:~/hw/hw3/mscha1

```
        for (i = 0; src[i] != '\0'; i++){
            dest[i+len] = src[i];
        }
        dest[i+len] = '\0';
        return dest;
    }

    static int strcmp(const char * left, const char * right){
        for (int i = 0; left[i] != '\0'; i++){
            if (left[i] - right[i] != 0)
                return left[i] - right[i];
        }
        return 0;
    }

    static int strncmp(const char *left, const char *right, int n){
        int i;
        for (i = 0; i < n; i++){
            if (left[i] - right[i] != 0)
                return left[i] - right[i];
        }
        return 0;
    }

    static char * strchr(char * str, int c){
        for (int i = 0; str[i] != '\0'; i++){
            if (str[i] == c)
                return str + i;
        }
        return nullptr;
    }

    static char * strstr(char * haystack, char * needle){
        int len = strlen(needle);
        char * s = haystack;
        char * p;
                                                                          35,6-13       18%
```

strcmp: compares left string to right string by subtracting the first two characters.

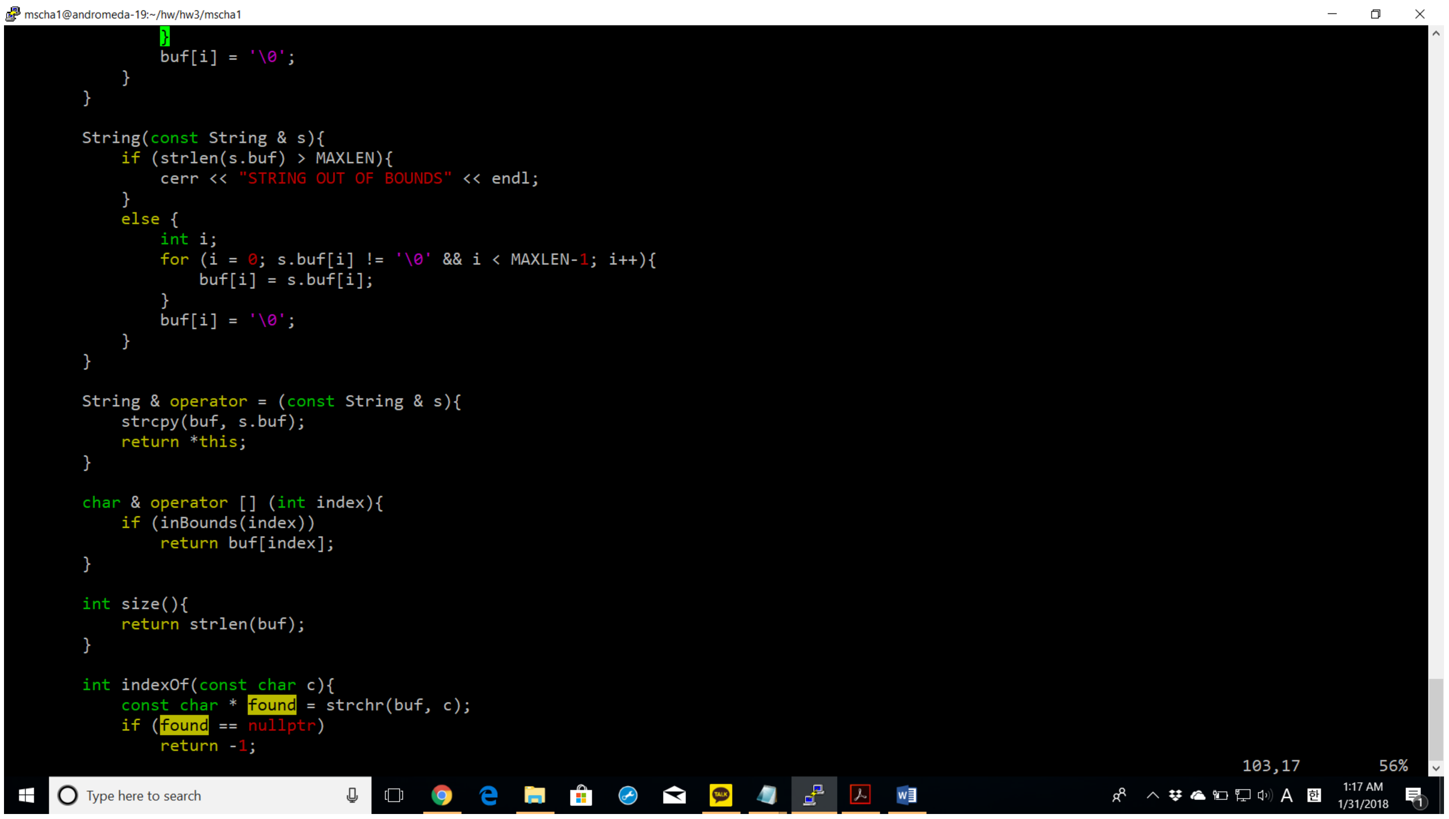


mscha1@andromeda-19:~/hw/hw3/mscha1

```
        char * s = haystack;
        char * p;
        while (* s != '\0'){
            p = strchr(s, needle[0]);
            if (p == nullptr)
                return nullptr;
            if (strncmp(p, needle, len) == 0)
                return p;
            else
                s = p  + 1;
        }
    }

    static const char * strstr(const char * haystack, const char * needle){
        return const_cast<const char *>(strstr(const_cast<char *>(haystack),
         const_cast<char *>(needle)));
    }

    static void reverse_cpy(char * dst, const char * src){
        int len = strlen(src);
        for (int i = len-1; i >= 0; --i)
            dst[len-i-1] = src[i];
        dst[len] = '\0';
    }

public:
    explicit String(const char * s = ""){
        if (strlen(s) > MAXLEN){
            cerr << "STRING OUT OF BOUNDS" << endl;
        }
        else {
            int i;
            for (i = 0; s[i] != '\0' && i < MAXLEN-1; i++){
                buf[i] = s[i];
            }
            buf[i] = '\0';
                                                                          69,6-13       37%
```

strstr: returns a pointer to the first index if needle is found inside haystack

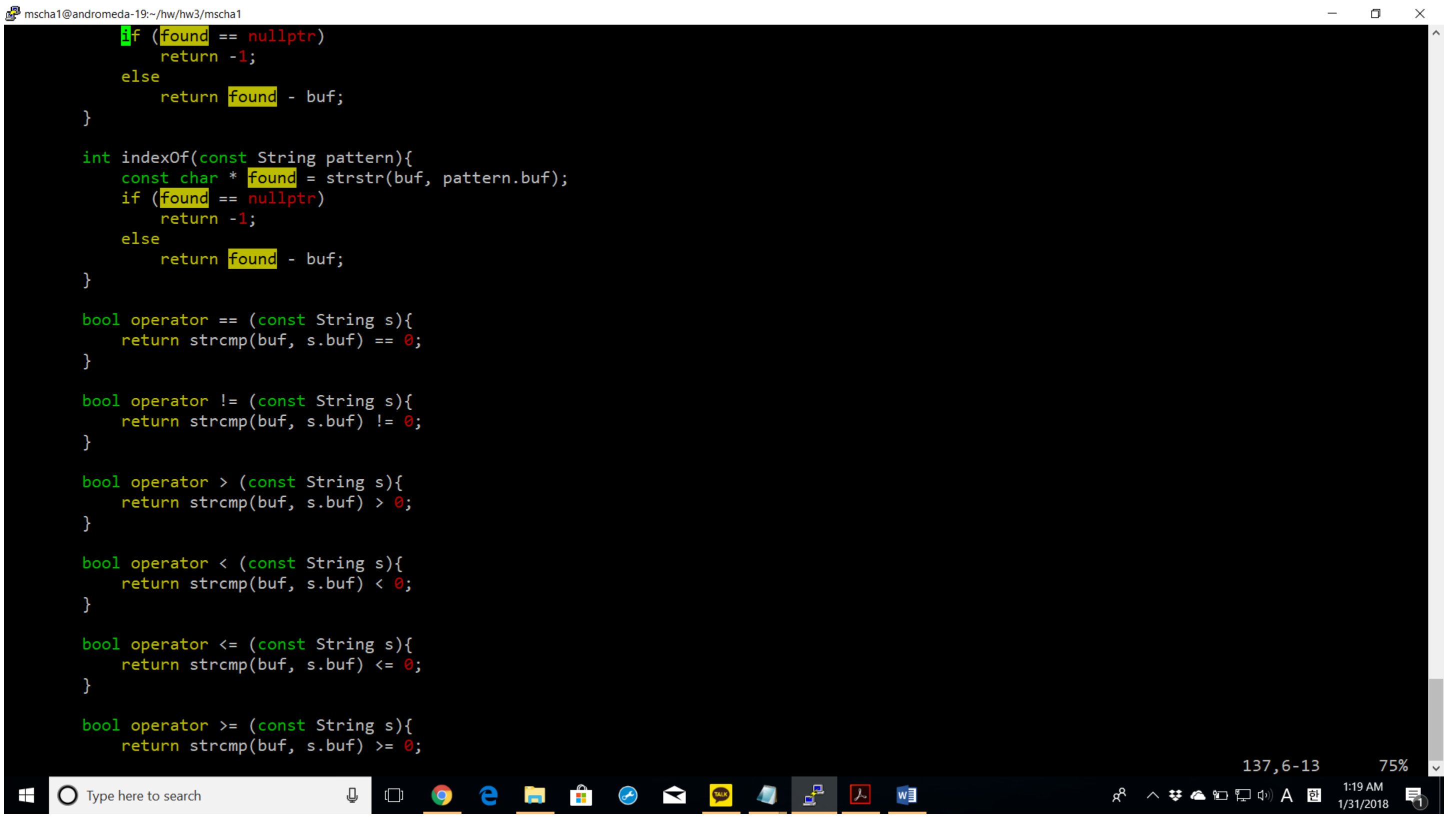
reverse_cpy: returns the opposite of the string by indexing the string backwards and storing it inside dest.



mscha1@andromeda-19:~/hw/hw3/mscha1

```
            }
            buf[i] = '\0';
        }
    }

    String(const String & s){
        if (strlen(s.buf) > MAXLEN){
            cerr << "STRING OUT OF BOUNDS" << endl;
        }
        else {
            int i;
            for (i = 0; s.buf[i] != '\0' && i < MAXLEN-1; i++){
                buf[i] = s.buf[i];
            }
            buf[i] = '\0';
        }
    }

    String & operator = (const String & s){
        strcpy(buf, s.buf);
        return *this;
    }

    char & operator [] (int index){
        if (inBounds(index))
            return buf[index];
    }

    int size(){
        return strlen(buf);
    }

    int indexOf(const char c){
        const char * found = strchr(buf, c);
        if (found == nullptr)
            return -1;
                                                                  103,17        56%
```

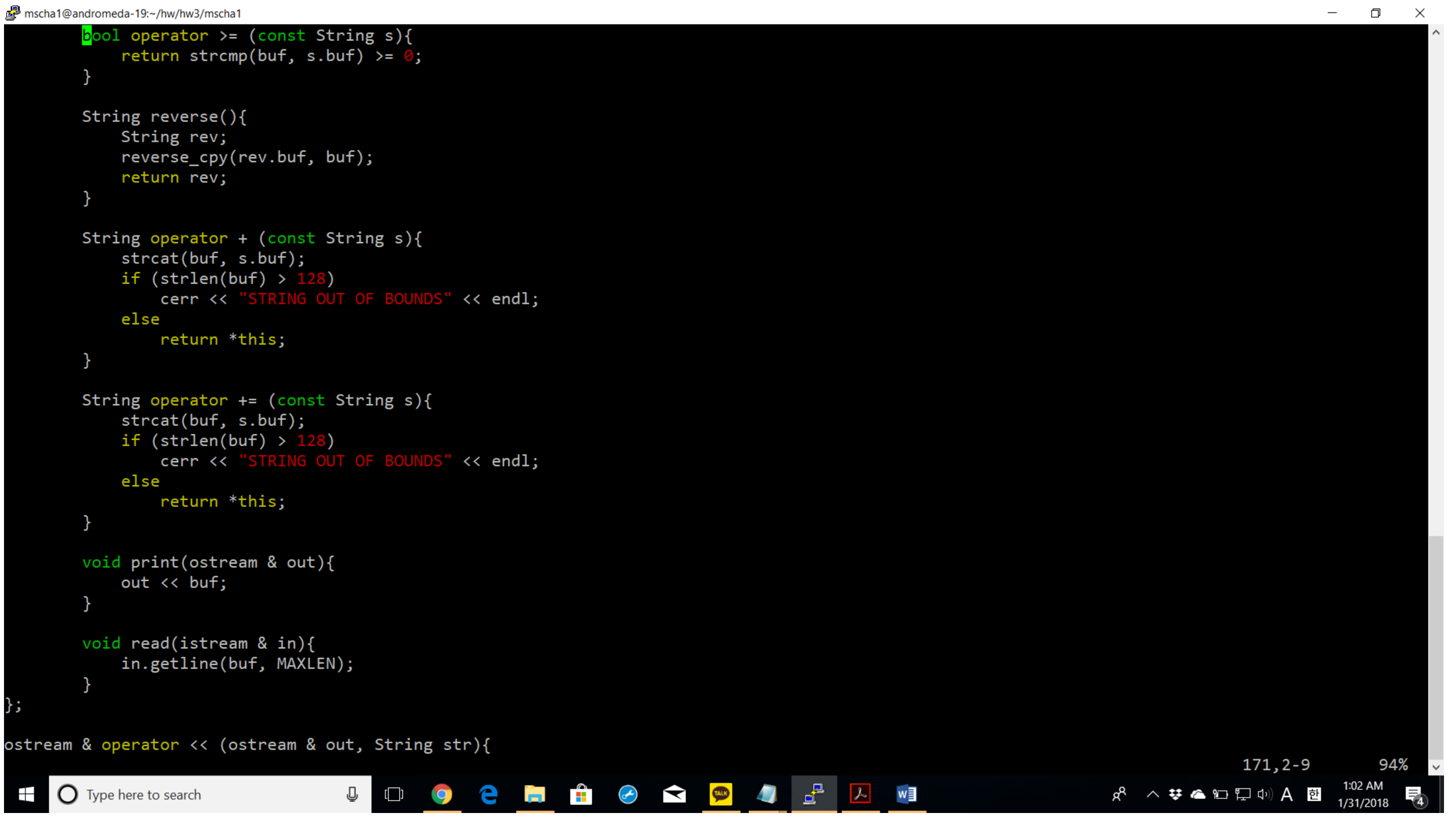
String(const String & s): constructs a string when a string is passed as an actual parameter. It places the null character at the end of the string.



mscha1@andromeda-19:~/hw/hw3/mscha1

```
        if (found == nullptr)
            return -1;
        else
            return found - buf;
    }

    int indexOf(const String pattern){
        const char * found = strstr(buf, pattern.buf);
        if (found == nullptr)
            return -1;
        else
            return found - buf;
    }

    bool operator == (const String s){
        return strcmp(buf, s.buf) == 0;
    }

    bool operator != (const String s){
        return strcmp(buf, s.buf) != 0;
    }

    bool operator > (const String s){
        return strcmp(buf, s.buf) > 0;
    }

    bool operator < (const String s){
        return strcmp(buf, s.buf) < 0;
    }

    bool operator <= (const String s){
        return strcmp(buf, s.buf) <= 0;
    }

    bool operator >= (const String s){
        return strcmp(buf, s.buf) >= 0;
                                                                  137,6-13      75%
```

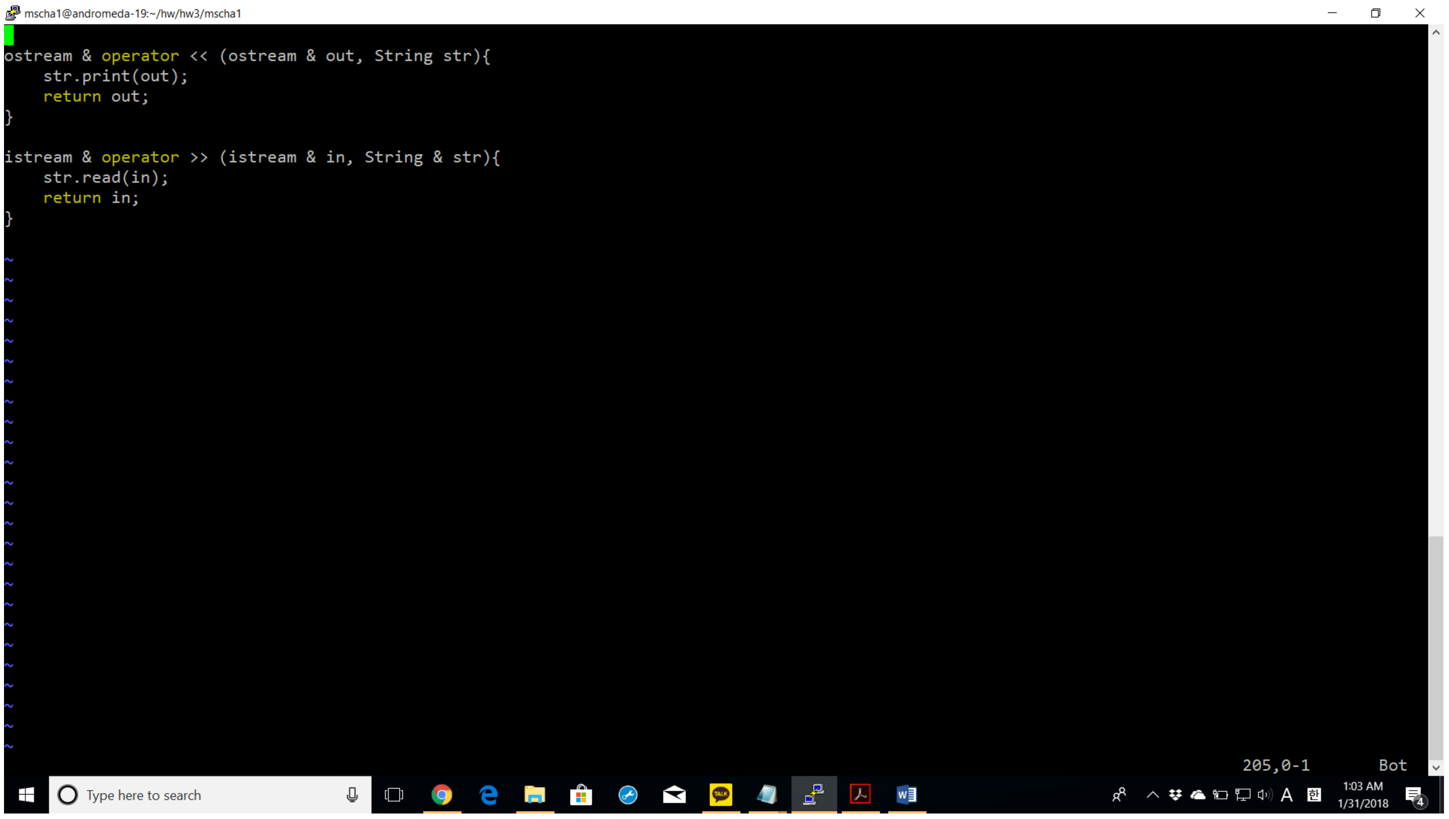
indexOf: uses the function strstr to get the pointer to the string pattern that matches to the current string and returns its index by subtracting the pointer minus the entire length of the buffer.



mscha1@andromeda-19:~/hw/hw3/mscha1

```
        bool operator >= (const String s){
            return strcmp(buf, s.buf) >= 0;
        }

        String reverse(){
            String rev;
            reverse_cpy(rev.buf, buf);
            return rev;
        }

        String operator + (const String s){
            strcat(buf, s.buf);
            if (strlen(buf) > 128)
                cerr << "STRING OUT OF BOUNDS" << endl;
            else
                return *this;
        }

        String operator += (const String s){
            strcat(buf, s.buf);
            if (strlen(buf) > 128)
                cerr << "STRING OUT OF BOUNDS" << endl;
            else
                return *this;
        }

        void print(ostream & out){
            out << buf;
        }

        void read(istream & in){
            in.getline(buf, MAXLEN);
        }
};

ostream & operator << (ostream & out, String str){
                                                                                                    171,2-9       94%
```

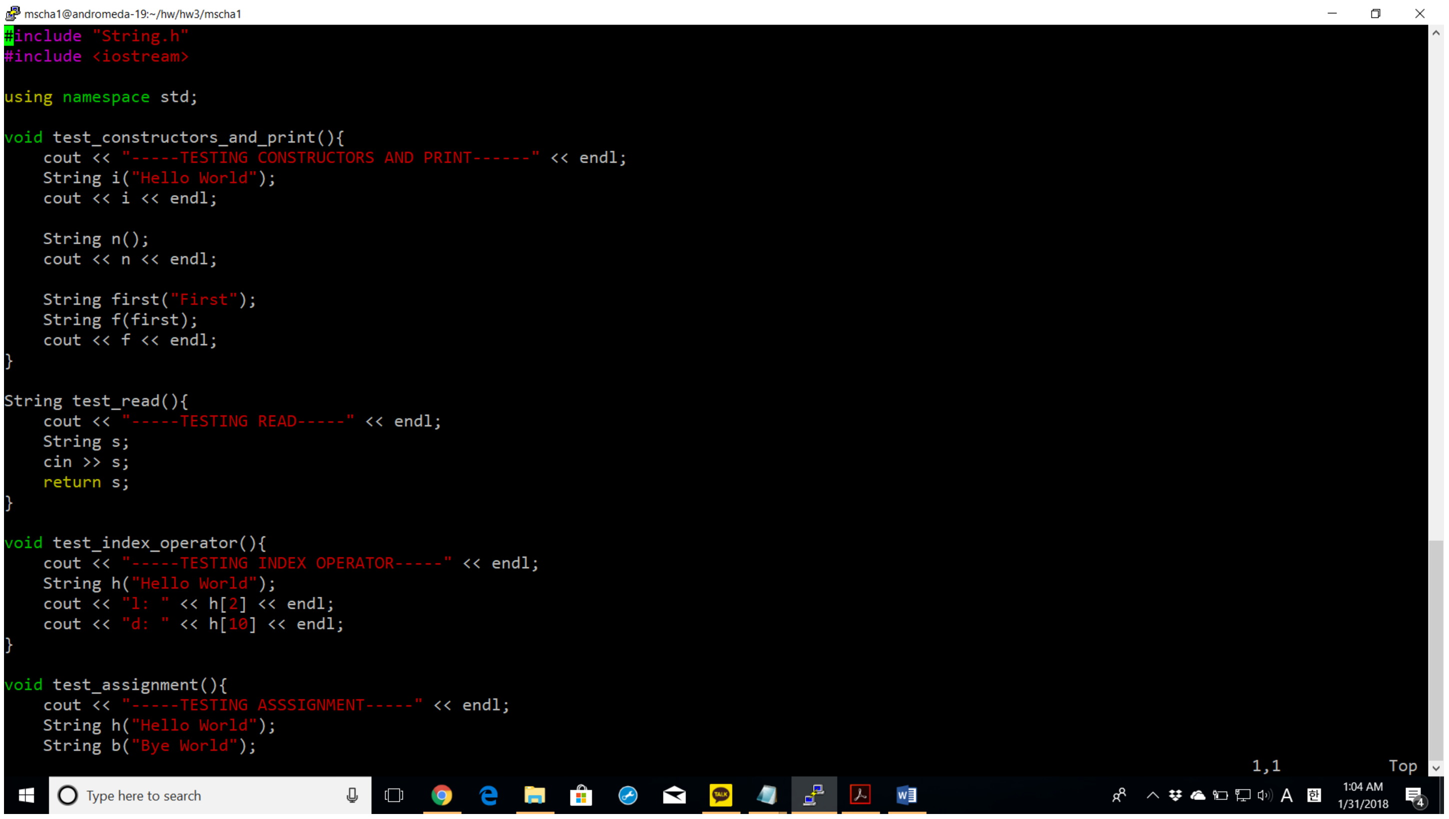
String operator + (const Strings): concatenates s into buf by using strcat.



mscha1@andromeda-19:~/hw/hw3/mscha1

```

ostream & operator << (ostream & out, String str){
    str.print(out);
    return out;
}

istream & operator >> (istream & in, String & str){
    str.read(in);
    return in;
}

~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
                                                                                                    205,0-1       Bot
```

string_test.cpp

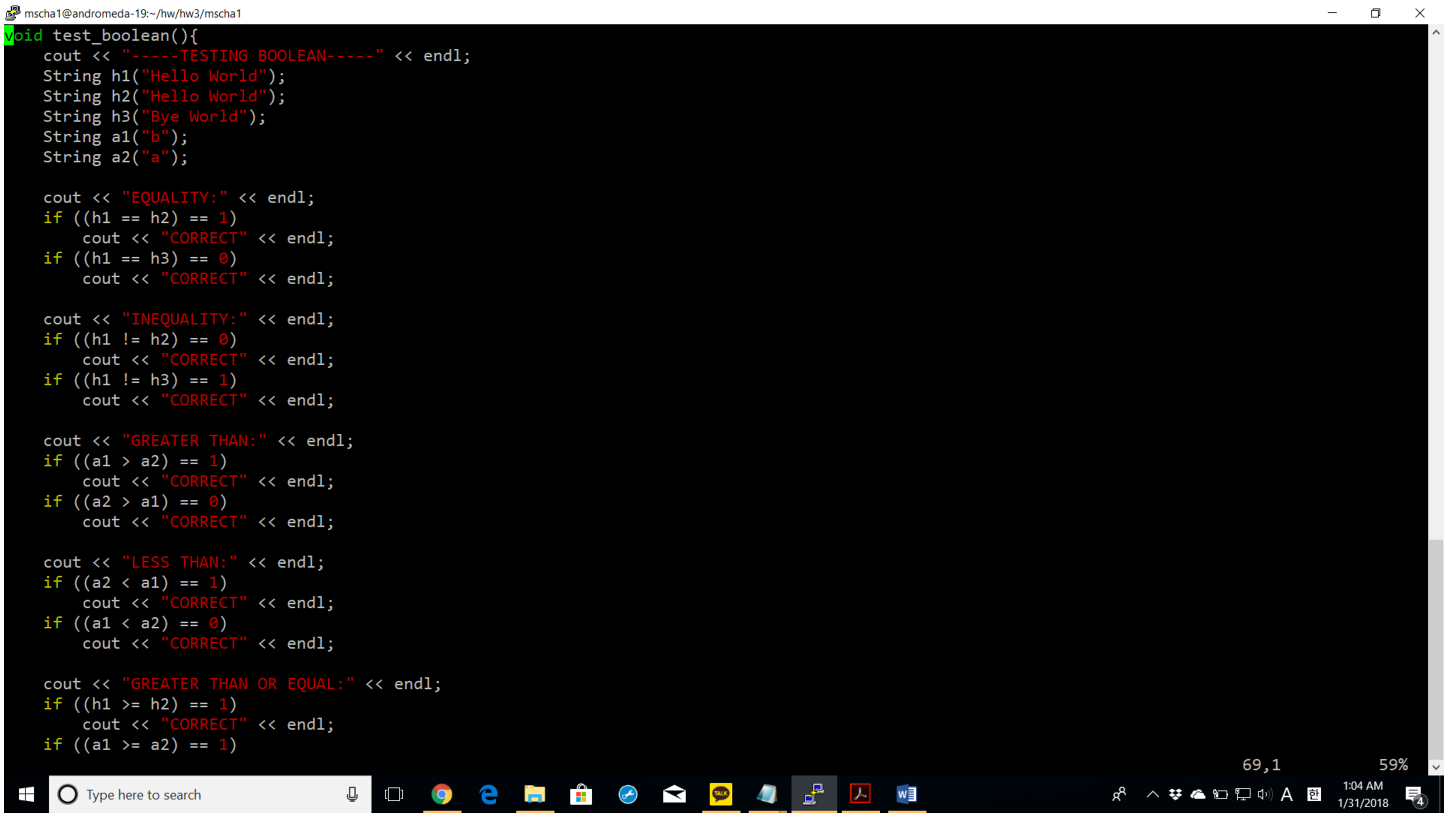


mscha1@andromeda-19:~/hw/hw3/mscha1

```
#include "String.h"
#include <iostream>

using namespace std;

void test_constructors_and_print(){
    cout << "-----TESTING CONSTRUCTORS AND PRINT------" << endl;
    String i("Hello World");
    cout << i << endl;

    String n();
    cout << n << endl;

    String first("First");
    String f(first);
    cout << f << endl;
}

String test_read(){
    cout << "-----TESTING READ-----" << endl;
    String s;
    cin >> s;
    return s;
}

void test_index_operator(){
    cout << "-----TESTING INDEX OPERATOR-----" << endl;
    String h("Hello World");
    cout << "l: " << h[2] << endl;
    cout << "d: " << h[10] << endl;
}

void test_assignment(){
    cout << "-----TESTING ASSIGNMENT-----" << endl;
    String h("Hello World");
    String b("Bye World");
                                                                                                         1,1           Top
```

Type here to search

1:04 AM
1/31/2018

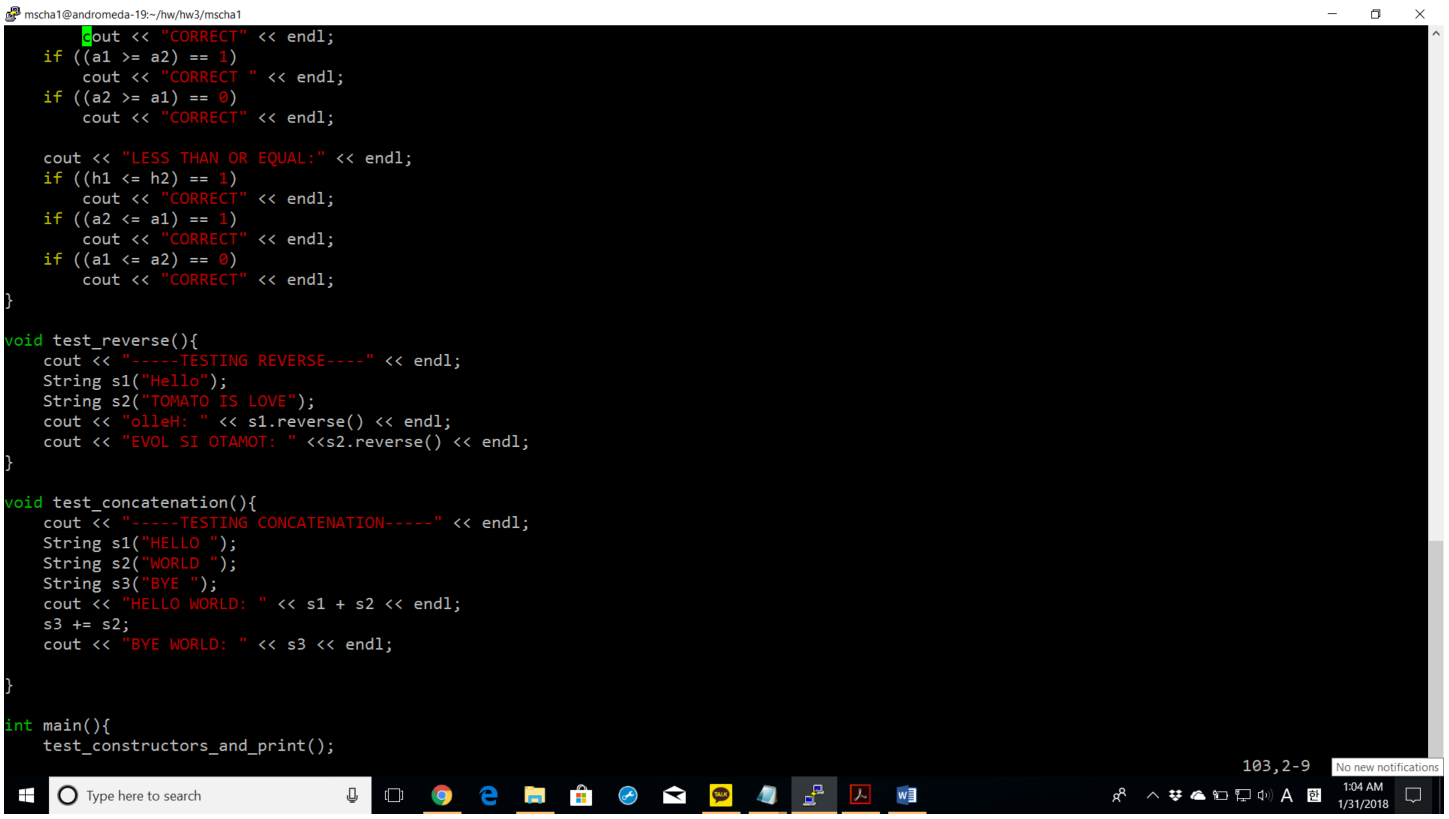


mscha1@andromeda-19:~/hw/hw3/mscha1

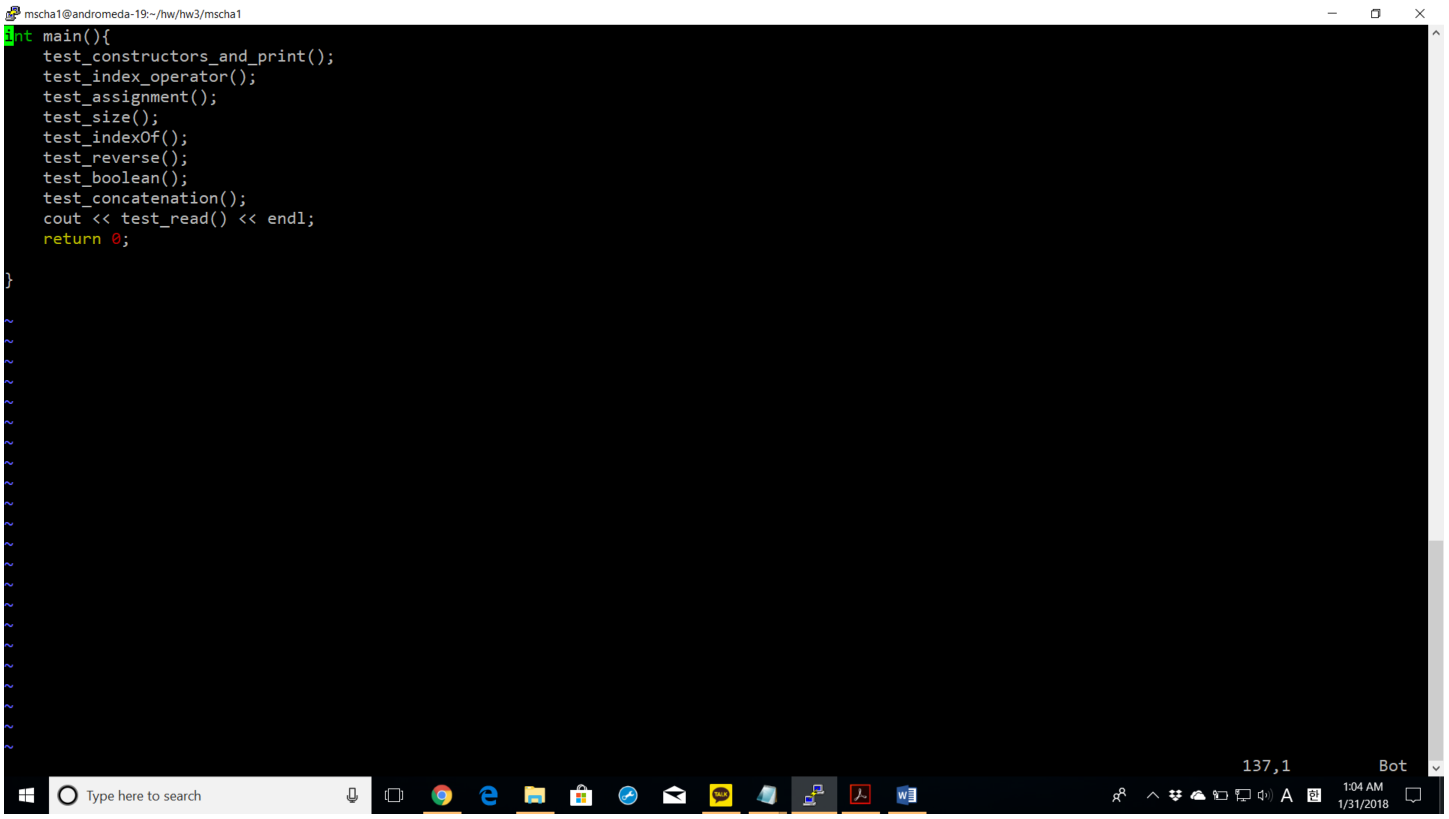
```
void test_boolean(){
    cout << "-----TESTING BOOLEAN-----" << endl;
    String h1("Hello World");
    String h2("Hello World");
    String h3("Bye World");
    String a1("b");
    String a2("a");

    cout << "EQUALITY:" << endl;
    if ((h1 == h2) == 1)
        cout << "CORRECT" << endl;
    if ((h1 == h3) == 0)
        cout << "CORRECT" << endl;

    cout << "INEQUALITY:" << endl;
    if ((h1 != h2) == 0)
        cout << "CORRECT" << endl;
    if ((h1 != h3) == 1)
        cout << "CORRECT" << endl;

    cout << "GREATER THAN:" << endl;
    if ((a1 > a2) == 1)
        cout << "CORRECT" << endl;
    if ((a2 > a1) == 0)
        cout << "CORRECT" << endl;

    cout << "LESS THAN:" << endl;
    if ((a2 < a1) == 1)
        cout << "CORRECT" << endl;
    if ((a1 < a2) == 0)
        cout << "CORRECT" << endl;

    cout << "GREATER THAN OR EQUAL:" << endl;
    if ((h1 >= h2) == 1)
        cout << "CORRECT" << endl;
    if ((a1 >= a2) == 1)
                                                                                                        69,1           59%
```

Type here to search

1:04 AM
1/31/2018

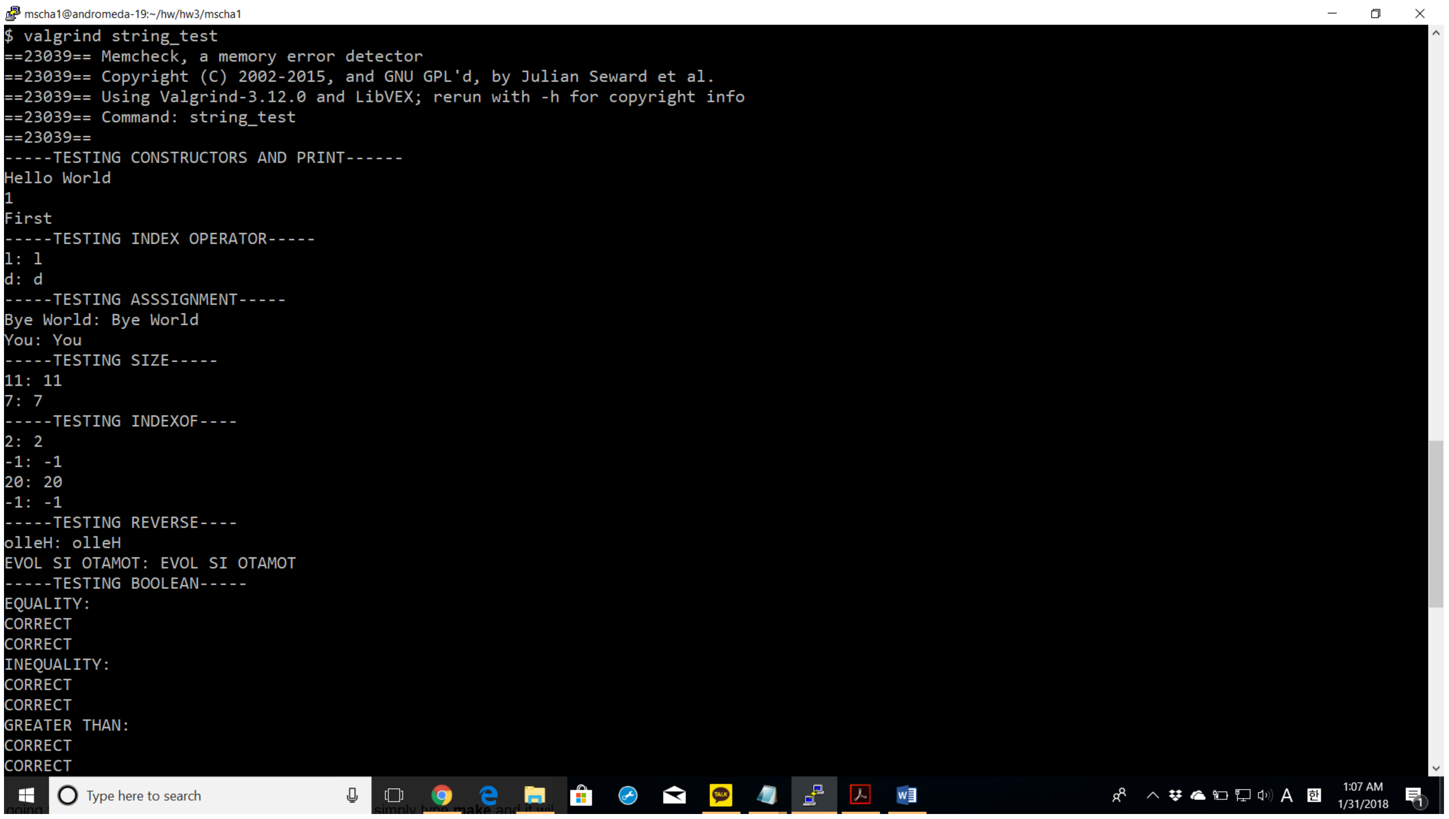


```
        cout << "CORRECT" << endl;
    if ((a1 >= a2) == 1)
        cout << "CORRECT " << endl;
    if ((a2 >= a1) == 0)
        cout << "CORRECT" << endl;

    cout << "LESS THAN OR EQUAL:" << endl;
    if ((h1 <= h2) == 1)
        cout << "CORRECT" << endl;
    if ((a2 <= a1) == 1)
        cout << "CORRECT" << endl;
    if ((a1 <= a2) == 0)
        cout << "CORRECT" << endl;
}

void test_reverse(){
    cout << "-----TESTING REVERSE----" << endl;
    String s1("Hello");
    String s2("TOMATO IS LOVE");
    cout << "olleH: " << s1.reverse() << endl;
    cout << "EVOL SI OTAMOT: " <<s2.reverse() << endl;
}

void test_concatenation(){
    cout << "-----TESTING CONCATENATION-----" << endl;
    String s1("HELLO ");
    String s2("WORLD ");
    String s3("BYE ");
    cout << "HELLO WORLD: " << s1 + s2 << endl;
    s3 += s2;
    cout << "BYE WORLD: " << s3 << endl;

}

int main(){
    test_constructors_and_print();
                                                                                        103,2-9
```

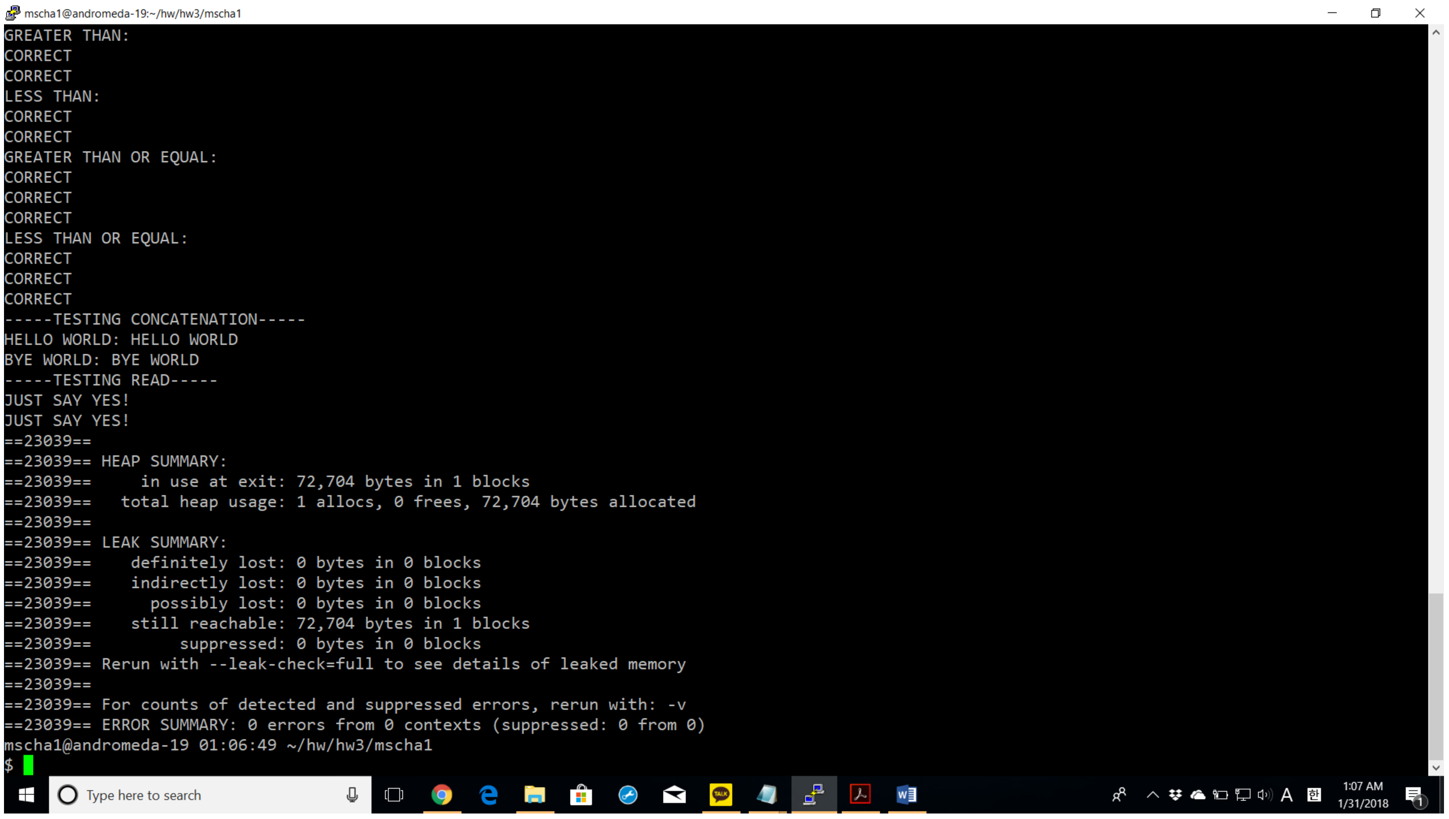


```
int main(){
    test_constructors_and_print();
    test_index_operator();
    test_assignment();
    test_size();
    test_indexOf();
    test_reverse();
    test_boolean();
    test_concatenation();
    cout << test_read() << endl;
    return 0;

}
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
                                                                                        137,1          Bot
```

Successful execution with Valgrind screenshot

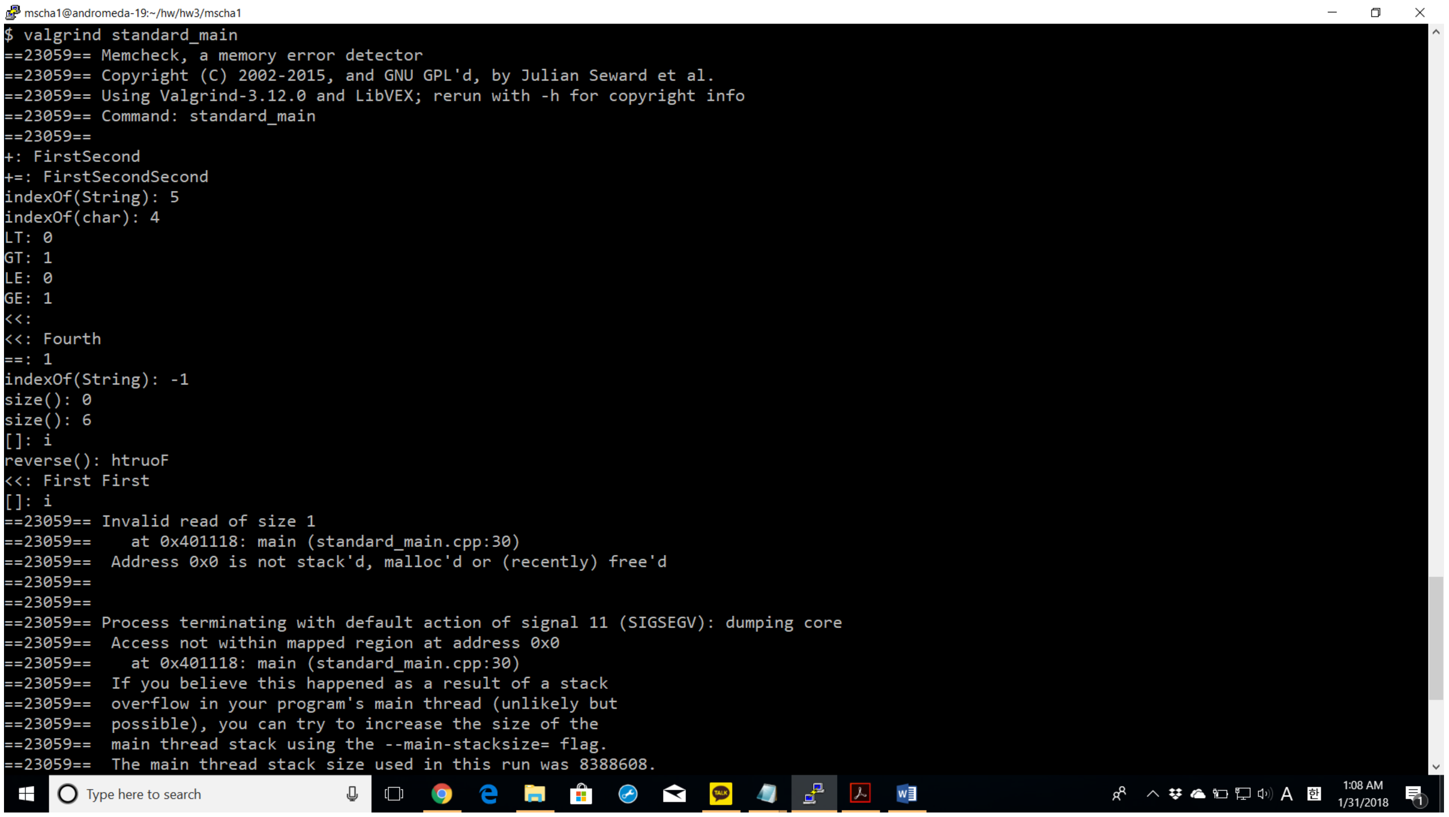

mscha1@andromeda-19:~/hw/hw3/mscha1

```
$ valgrind string_test
==23039== Memcheck, a memory error detector
==23039== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
==23039== Using Valgrind-3.12.0 and LibVEX; rerun with -h for copyright info
==23039== Command: string_test
==23039==
-----TESTING CONSTRUCTORS AND PRINT------
Hello World
1
First
-----TESTING INDEX OPERATOR-----
l: l
d: d
-----TESTING ASSSIGNMENT-----
Bye World: Bye World
You: You
-----TESTING SIZE-----
11: 11
7: 7
-----TESTING INDEXOF----
2: 2
-1: -1
20: 20
-1: -1
-----TESTING REVERSE----
olleH: olleH
EVOL SI OTAMOT: EVOL SI OTAMOT
-----TESTING BOOLEAN-----
EQUALITY:
CORRECT
CORRECT
INEQUALITY:
CORRECT
CORRECT
GREATER THAN:
CORRECT
CORRECT
```


mscha1@andromeda-19:~/hw/hw3/mscha1

```
GREATER THAN:
CORRECT
CORRECT
LESS THAN:
CORRECT
CORRECT
GREATER THAN OR EQUAL:
CORRECT
CORRECT
CORRECT
LESS THAN OR EQUAL:
CORRECT
CORRECT
CORRECT
-----TESTING CONCATENATION-----
HELLO WORLD: HELLO WORLD
BYE WORLD: BYE WORLD
-----TESTING READ-----
JUST SAY YES!
JUST SAY YES!
==23039==
==23039== HEAP SUMMARY:
==23039==     in use at exit: 72,704 bytes in 1 blocks
==23039==   total heap usage: 1 allocs, 0 frees, 72,704 bytes allocated
==23039==
==23039== LEAK SUMMARY:
==23039==    definitely lost: 0 bytes in 0 blocks
==23039==    indirectly lost: 0 bytes in 0 blocks
==23039==      possibly lost: 0 bytes in 0 blocks
==23039==    still reachable: 72,704 bytes in 1 blocks
==23039==         suppressed: 0 bytes in 0 blocks
==23039== Rerun with --leak-check=full to see details of leaked memory
==23039==
==23039== For counts of detected and suppressed errors, rerun with: -v
==23039== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
mscha1@andromeda-19 01:06:49 ~/hw/hw3/mscha1
$
```

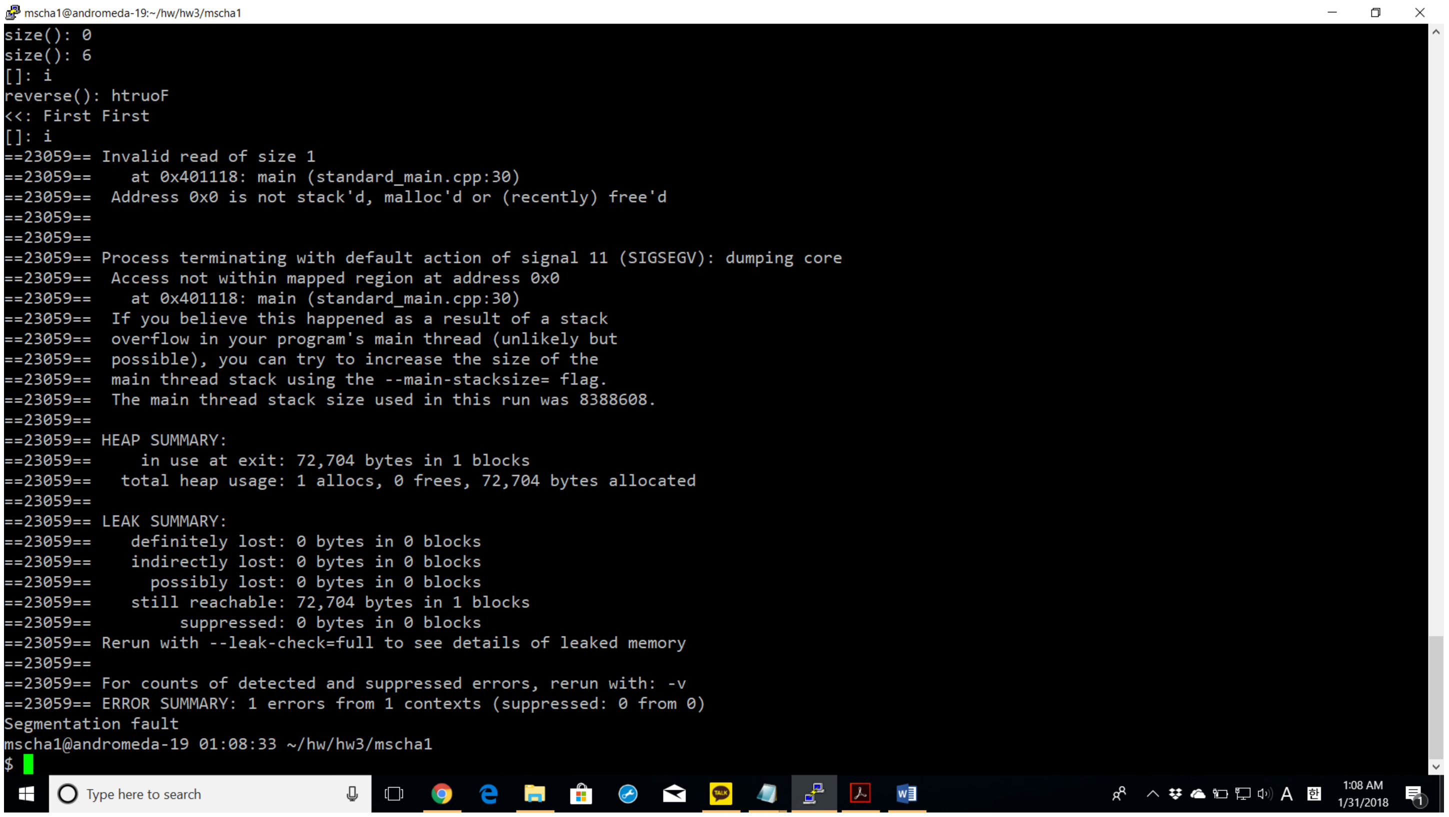
standard_main.cpp



mscha1@andromeda-19:~/hw/hw3/mscha1

```
$ valgrind standard_main
==23059== Memcheck, a memory error detector
==23059== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
==23059== Using Valgrind-3.12.0 and LibVEX; rerun with -h for copyright info
==23059== Command: standard_main
==23059==
+: FirstSecond
+=: FirstSecondSecond
indexOf(String): 5
indexOf(char): 4
LT: 0
GT: 1
LE: 0
GE: 1
<<:
<<: Fourth
==: 1
indexOf(String): -1
size(): 0
size(): 6
[]: i
reverse(): htruoF
<<: First First
[]: i
==23059== Invalid read of size 1
==23059==    at 0x401118: main (standard_main.cpp:30)
==23059==  Address 0x0 is not stack'd, malloc'd or (recently) free'd
==23059==
==23059==
==23059== Process terminating with default action of signal 11 (SIGSEGV): dumping core
==23059==  Access not within mapped region at address 0x0
==23059==    at 0x401118: main (standard_main.cpp:30)
==23059==  If you believe this happened as a result of a stack
==23059==  overflow in your program's main thread (unlikely but
==23059==  possible), you can try to increase the size of the
==23059==  main thread stack using the --main-stacksize= flag.
==23059==  The main thread stack size used in this run was 8388608.
```

Type here to search 1:08 AM 1/31/2018

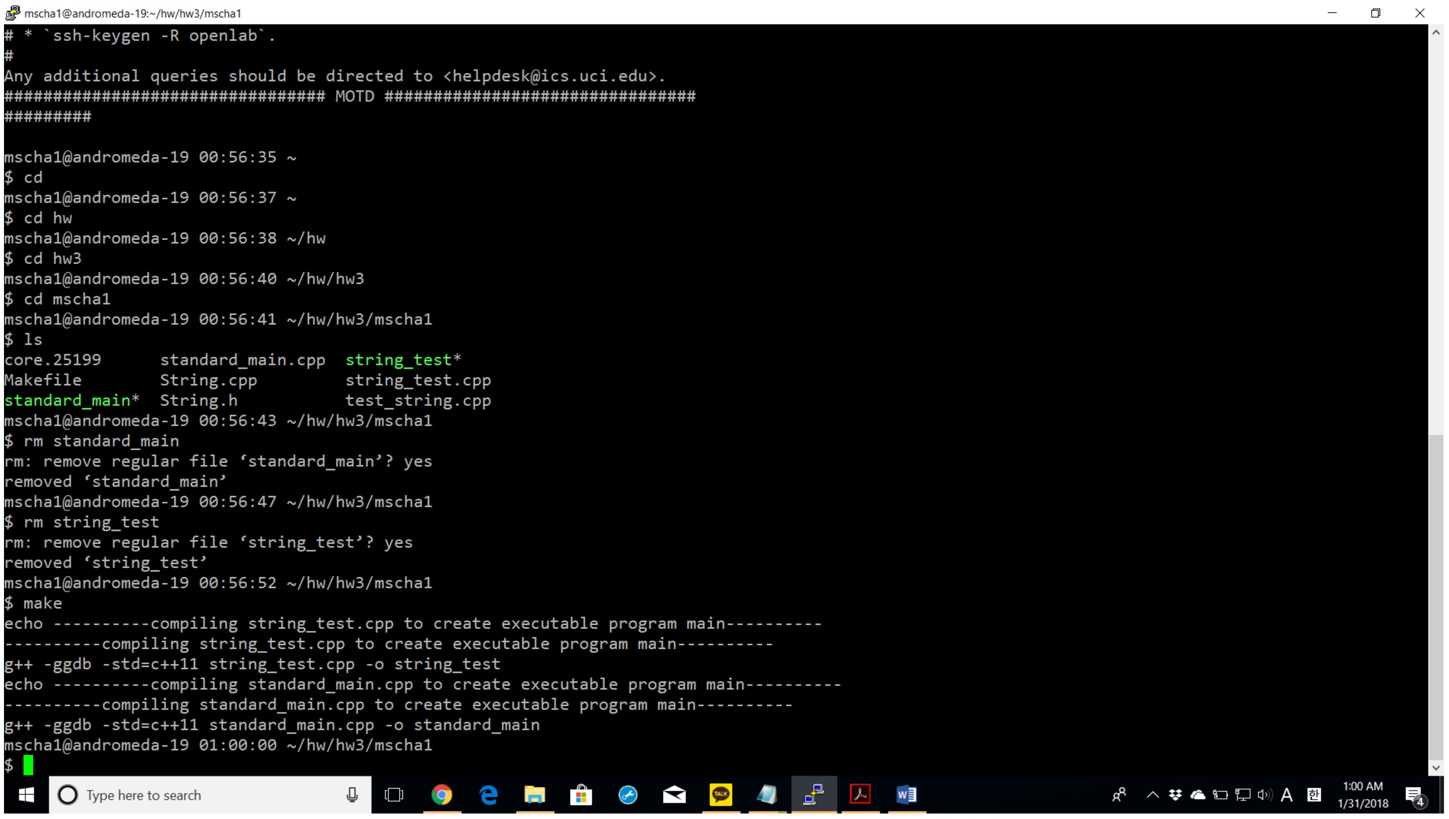


mscha1@andromeda-19:~/hw/hw3/mscha1

```
size(): 0
size(): 6
[]: i
reverse(): htruoF
<<: First First
[]: i
==23059== Invalid read of size 1
==23059==    at 0x401118: main (standard_main.cpp:30)
==23059==  Address 0x0 is not stack'd, malloc'd or (recently) free'd
==23059==
==23059==
==23059== Process terminating with default action of signal 11 (SIGSEGV): dumping core
==23059==  Access not within mapped region at address 0x0
==23059==    at 0x401118: main (standard_main.cpp:30)
==23059==  If you believe this happened as a result of a stack
==23059==  overflow in your program's main thread (unlikely but
==23059==  possible), you can try to increase the size of the
==23059==  main thread stack using the --main-stacksize= flag.
==23059==  The main thread stack size used in this run was 8388608.
==23059==
==23059== HEAP SUMMARY:
==23059==     in use at exit: 72,704 bytes in 1 blocks
==23059==   total heap usage: 1 allocs, 0 frees, 72,704 bytes allocated
==23059==
==23059== LEAK SUMMARY:
==23059==    definitely lost: 0 bytes in 0 blocks
==23059==    indirectly lost: 0 bytes in 0 blocks
==23059==      possibly lost: 0 bytes in 0 blocks
==23059==    still reachable: 72,704 bytes in 1 blocks
==23059==         suppressed: 0 bytes in 0 blocks
==23059== Rerun with --leak-check=full to see details of leaked memory
==23059==
==23059== For counts of detected and suppressed errors, rerun with: -v
==23059== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
Segmentation fault
mscha1@andromeda-19 01:08:33 ~/hw/hw3/mscha1
$
```

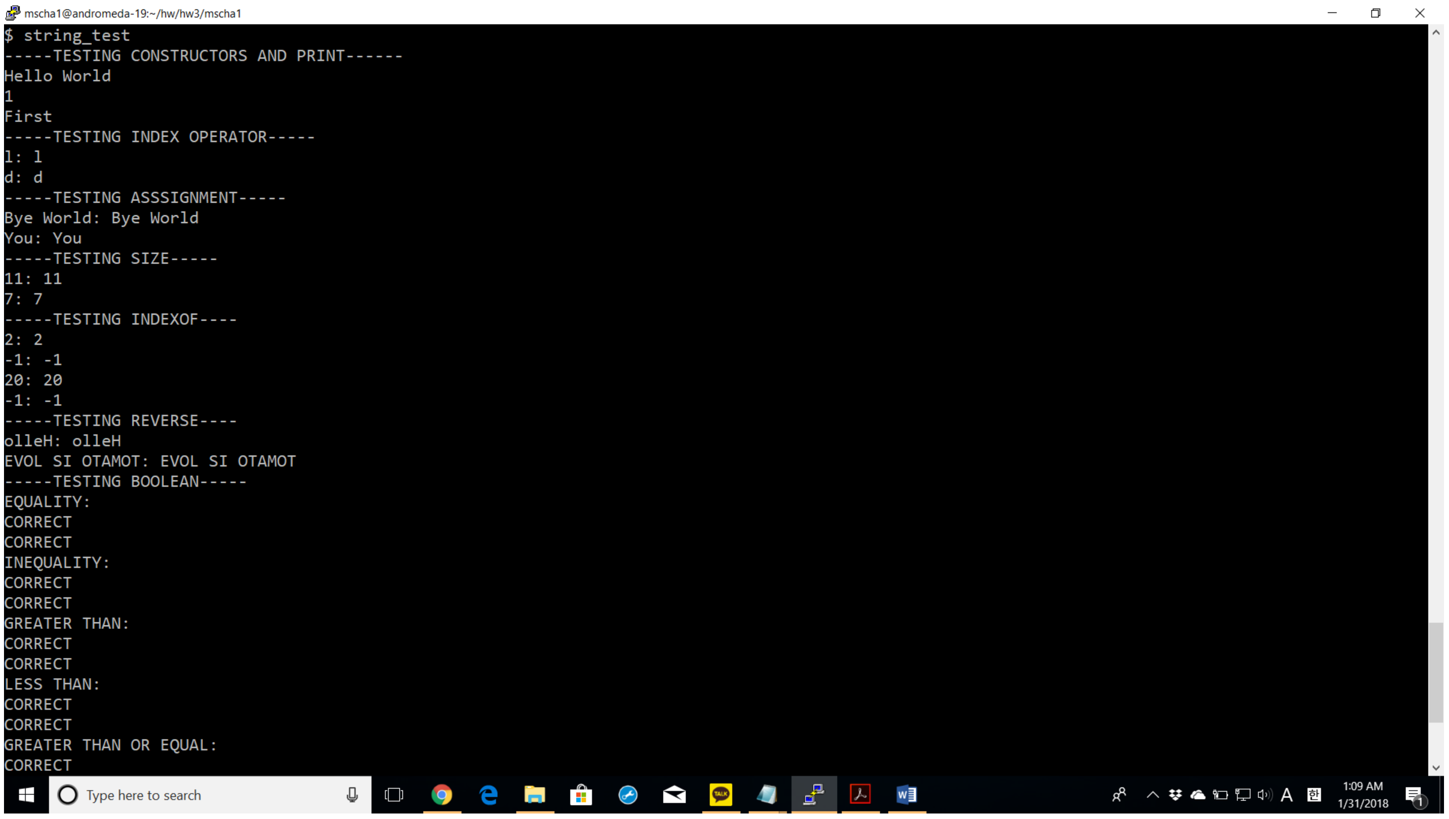
Type here to search 1:08 AM 1/31/2018

Test Output screenshots with descriptions of each test



```
# * `ssh-keygen -R openlab`.
#
Any additional queries should be directed to <helpdesk@ics.uci.edu>.
################################# MOTD ################################
#########

mscha1@andromeda-19 00:56:35 ~
$ cd
mscha1@andromeda-19 00:56:37 ~
$ cd hw
mscha1@andromeda-19 00:56:38 ~/hw
$ cd hw3
mscha1@andromeda-19 00:56:40 ~/hw/hw3
$ cd mscha1
mscha1@andromeda-19 00:56:41 ~/hw/hw3/mscha1
$ ls
core.25199      standard_main.cpp  string_test*
Makefile        String.cpp         string_test.cpp
standard_main*  String.h           test_string.cpp
mscha1@andromeda-19 00:56:43 ~/hw/hw3/mscha1
$ rm standard_main
rm: remove regular file ‘standard_main’? yes
removed ‘standard_main’
mscha1@andromeda-19 00:56:47 ~/hw/hw3/mscha1
$ rm string_test
rm: remove regular file ‘string_test’? yes
removed ‘string_test’
mscha1@andromeda-19 00:56:52 ~/hw/hw3/mscha1
$ make
echo ----------compiling string_test.cpp to create executable program main----------
----------compiling string_test.cpp to create executable program main----------
g++ -ggdb -std=c++11 string_test.cpp -o string_test
echo ----------compiling standard_main.cpp to create executable program main----------
----------compiling standard_main.cpp to create executable program main----------
g++ -ggdb -std=c++11 standard_main.cpp -o standard_main
mscha1@andromeda-19 01:00:00 ~/hw/hw3/mscha1
$
```

string_test.cpp



```
$ string_test
-----TESTING CONSTRUCTORS AND PRINT------
Hello World
1
First
-----TESTING INDEX OPERATOR-----
l: l
d: d
-----TESTING ASSSIGNMENT-----
Bye World: Bye World
You: You
-----TESTING SIZE-----
11: 11
7: 7
-----TESTING INDEXOF----
2: 2
-1: -1
20: 20
-1: -1
-----TESTING REVERSE----
olleH: olleH
EVOL SI OTAMOT: EVOL SI OTAMOT
-----TESTING BOOLEAN-----
EQUALITY:
CORRECT
CORRECT
INEQUALITY:
CORRECT
CORRECT
GREATER THAN:
CORRECT
CORRECT
LESS THAN:
CORRECT
CORRECT
GREATER THAN OR EQUAL:
CORRECT
```

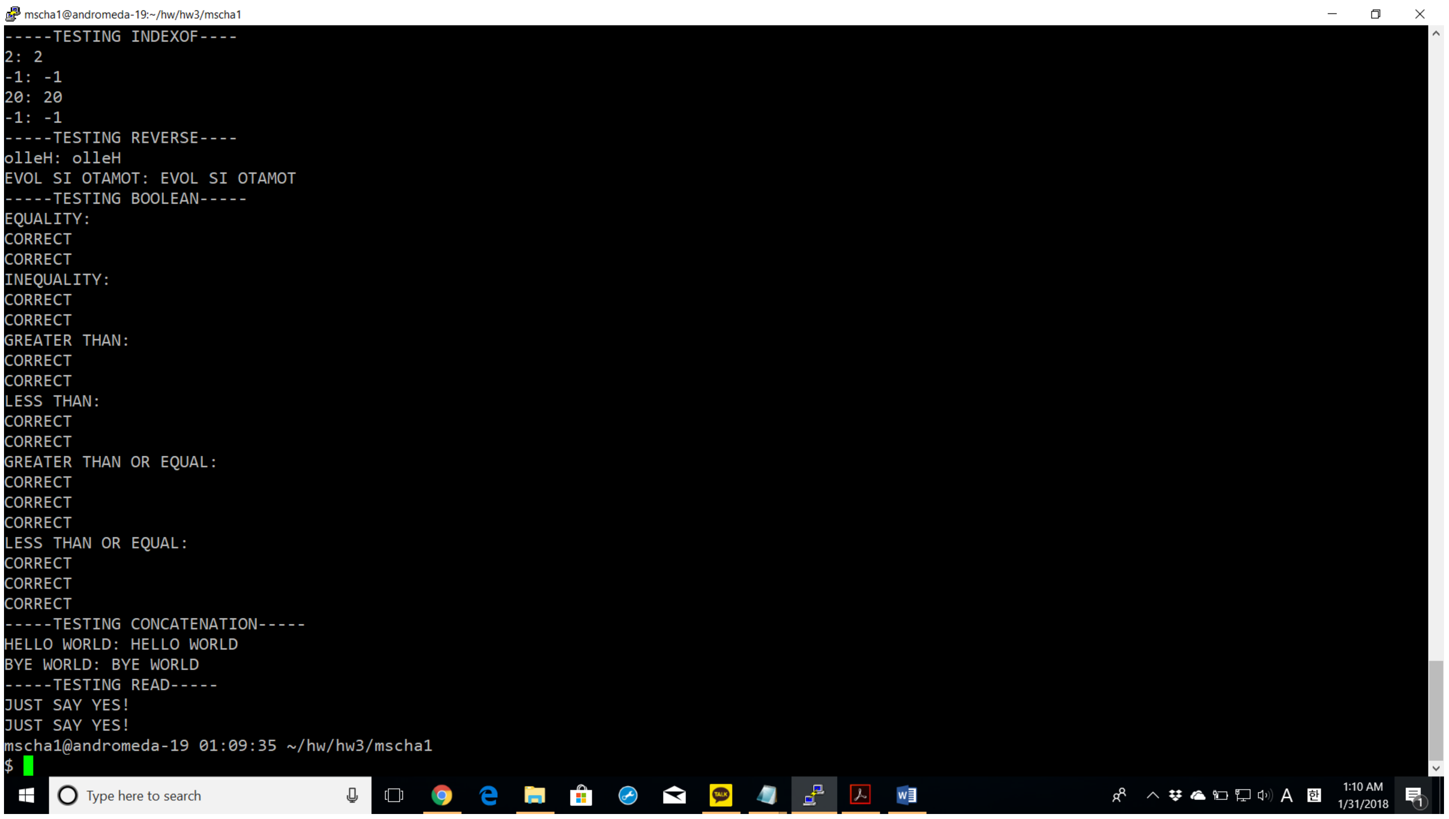
test_constructors_and_print: constructs the strings l, n, and f and outputs them to the terminal.

test_index_operator: tested if overloading the index operator resulted in the expected output.

test_size: tested if the size of the string corresponded to the expected output.

test_indexOf: tests if the calling the function indexOf corresponds to the expected output.

test_boolean: tested the Boolean functions by displaying correct if the expected output was obtained from the function in the terminal.



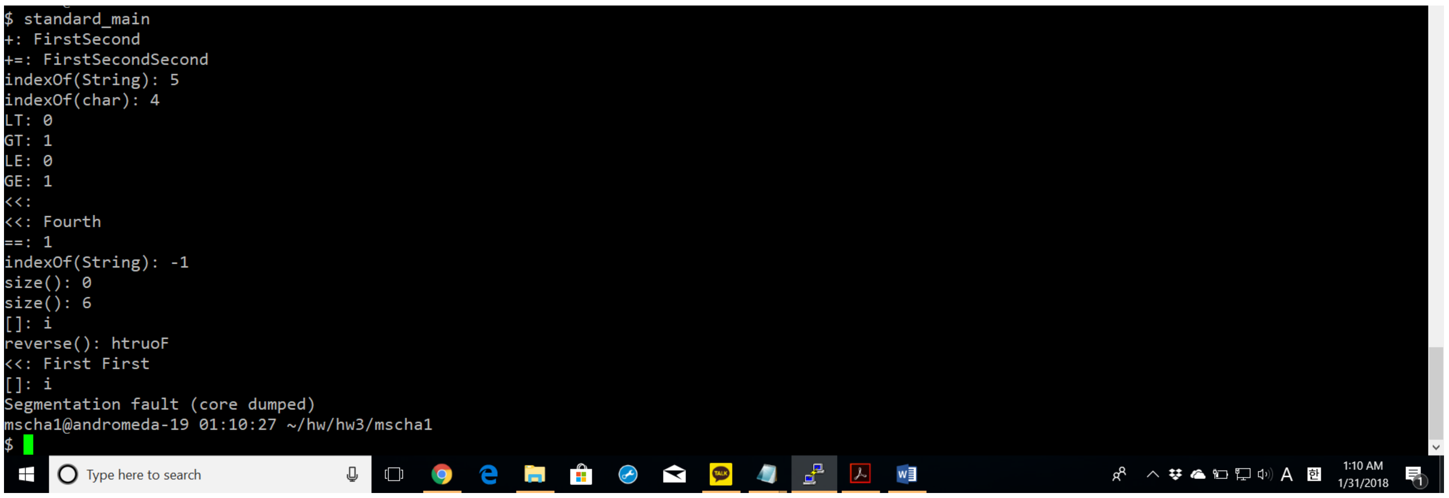
mscha1@andromeda-19:~/hw/hw3/mscha1

```
-----TESTING INDEXOF----
2: 2
-1: -1
20: 20
-1: -1
-----TESTING REVERSE----
olleH: olleH
EVOL SI OTAMOT: EVOL SI OTAMOT
-----TESTING BOOLEAN-----
EQUALITY:
CORRECT
CORRECT
INEQUALITY:
CORRECT
CORRECT
GREATER THAN:
CORRECT
CORRECT
LESS THAN:
CORRECT
CORRECT
GREATER THAN OR EQUAL:
CORRECT
CORRECT
CORRECT
LESS THAN OR EQUAL:
CORRECT
CORRECT
CORRECT
-----TESTING CONCATENATION-----
HELLO WORLD: HELLO WORLD
BYE WORLD: BYE WORLD
-----TESTING READ-----
JUST SAY YES!
JUST SAY YES!
mscha1@andromeda-19 01:09:35 ~/hw/hw3/mscha1
$ 
```

test_concatenation: tested if concatenating two strings together resulted in the expected output on the left. E.g. HELLO + WORLD = HELLO WORLD

test_read: tested if the corresponding variable is stored in buf and outputted in the terminal when the user inputs a string.

standard_main.cpp



```
$ standard_main
+: FirstSecond
+=: FirstSecondSecond
indexOf(String): 5
indexOf(char): 4
LT: 0
GT: 1
LE: 0
GE: 1
<<:
<<: Fourth
==: 1
indexOf(String): -1
size(): 0
size(): 6
[]: i
reverse(): htruoF
<<: First First
[]: i
Segmentation fault (core dumped)
mscha1@andromeda-19 01:10:27 ~/hw/hw3/mscha1
$ 
```