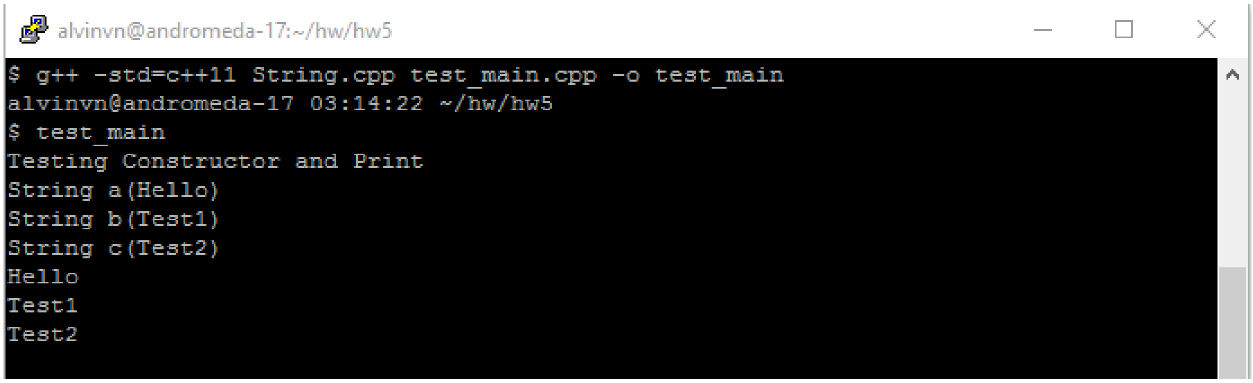
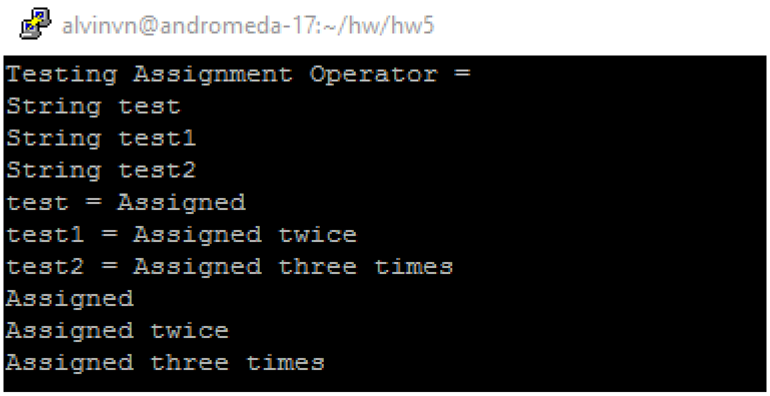
TEST_MAIN

CONSTRUCTOR:



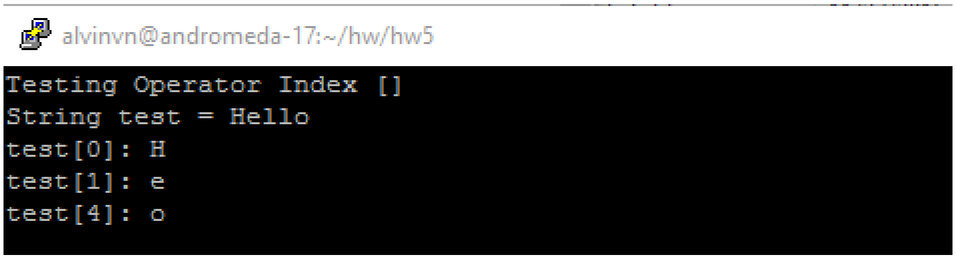
```
$ g++ -std=c++11 String.cpp test_main.cpp -o test_main
alvinvn@andromeda-17 03:14:22 ~/hw/hw5
$ test_main
Testing Constructor and Print
String a(Hello)
String b(Test1)
String c(Test2)
Hello
Test1
Test2
```

ASSIGNMENT OPERATOR =



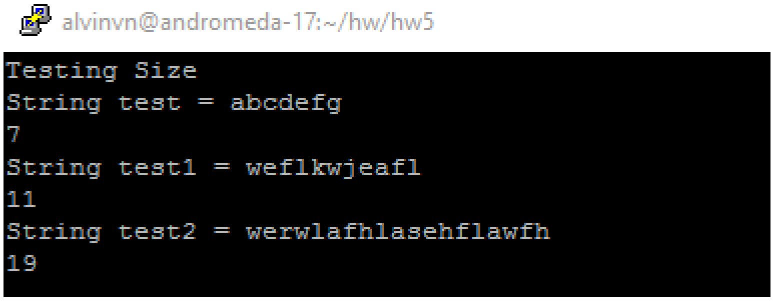
```
Testing Assignment Operator =
String test
String test1
String test2
test = Assigned
test1 = Assigned twice
test2 = Assigned three times
Assigned
Assigned twice
Assigned three times
```

OPERATOR INDEX []



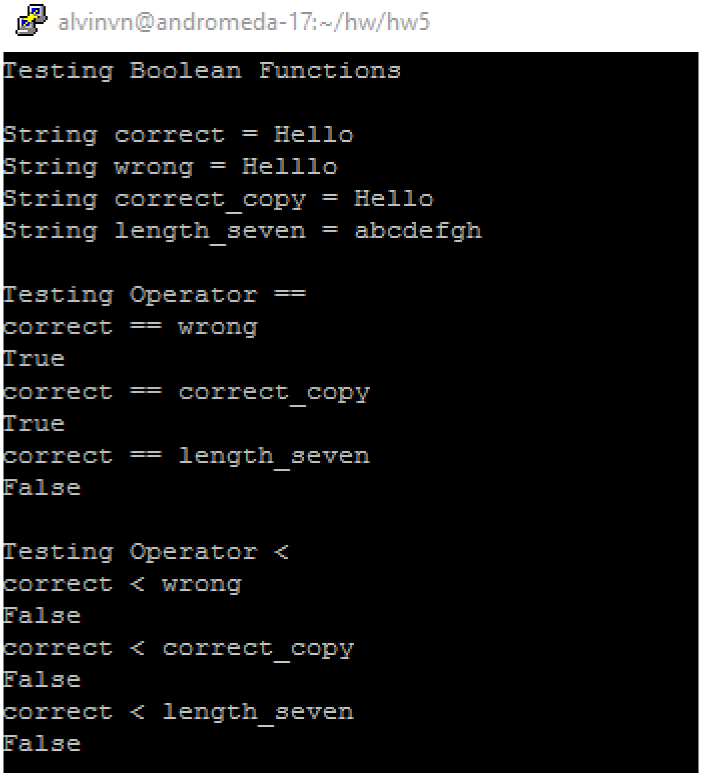
```
Testing Operator Index []
String test = Hello
test[0]: H
test[1]: e
test[4]: o
```

SIZE



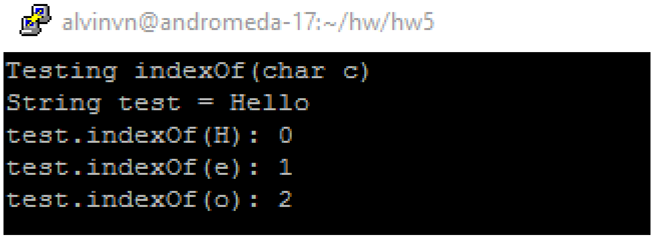
```
Testing Size
String test = abcdefg
7
String test1 = weflkwjeafl
11
String test2 = werwlafhlasehflawfh
19
```

BOOLEAN FUNCTIONS == AND <



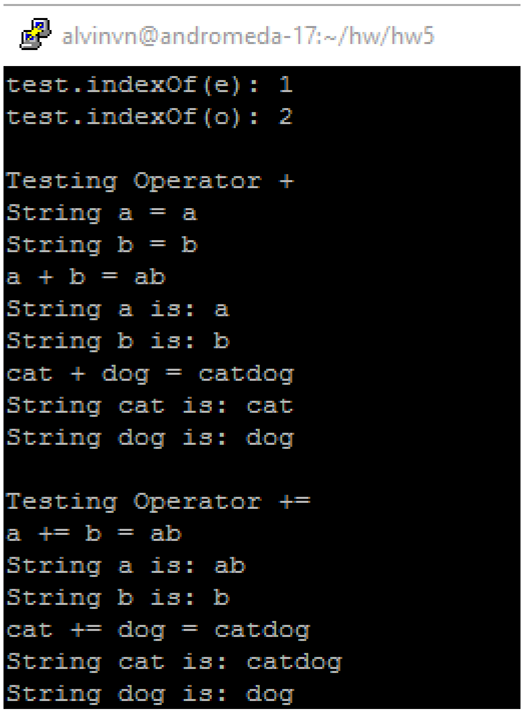
```
Testing Boolean Functions

String correct = Hello
String wrong = Helllo
String correct_copy = Hello
String length_seven = abcdefgh

Testing Operator ==
correct == wrong
True
correct == correct_copy
True
correct == length_seven
False

Testing Operator <
correct < wrong
False
correct < correct_copy
False
correct < length_seven
False
```

INDEXOF



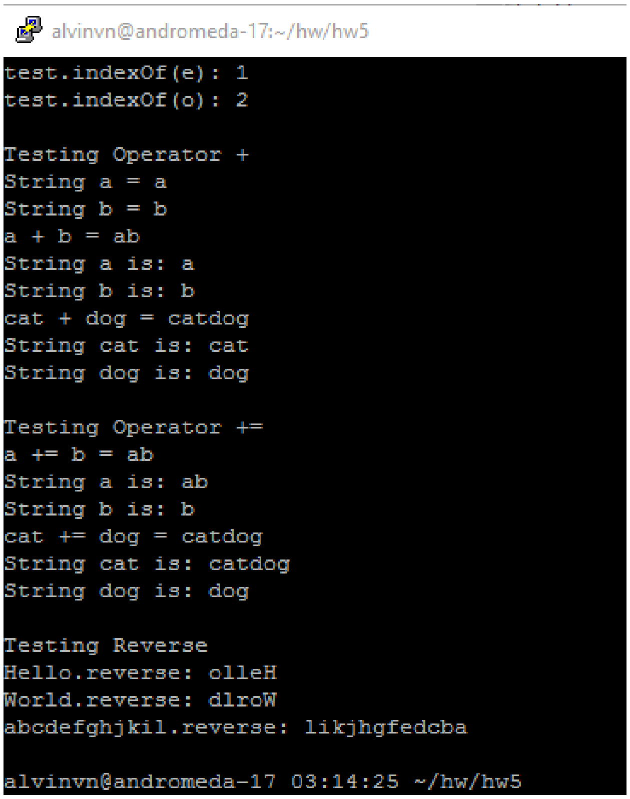
```
Testing indexOf(char c)
String test = Hello
test.indexOf(H): 0
test.indexOf(e): 1
test.indexOf(o): 2
```

ADDITION + AND +=



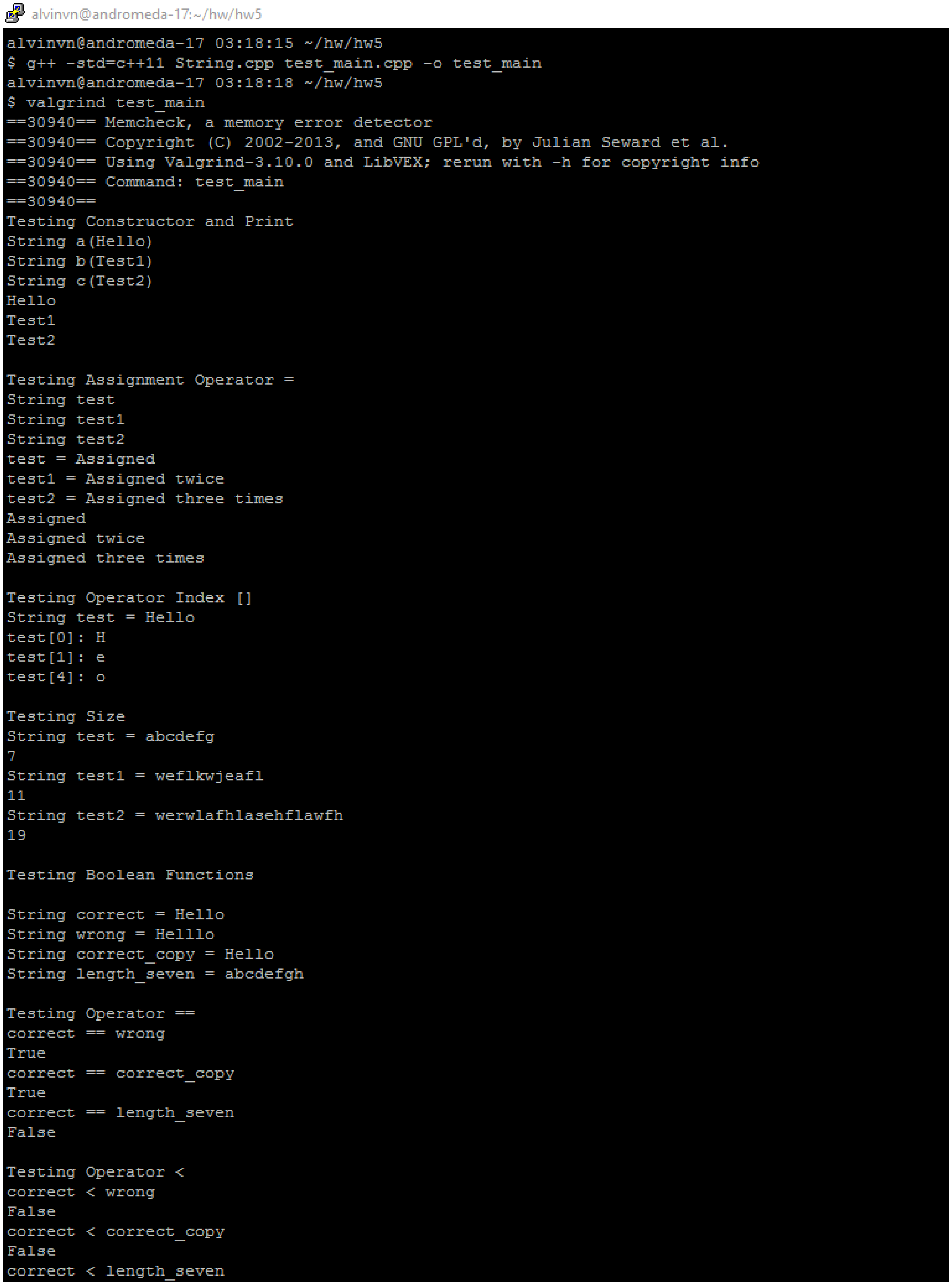
```
test.indexOf(e): 1
test.indexOf(o): 2

Testing Operator +
String a = a
String b = b
a + b = ab
String a is: a
String b is: b
cat + dog = catdog
String cat is: cat
String dog is: dog

Testing Operator +=
a += b = ab
String a is: ab
String b is: b
cat += dog = catdog
String cat is: catdog
String dog is: dog
```

REVERSE



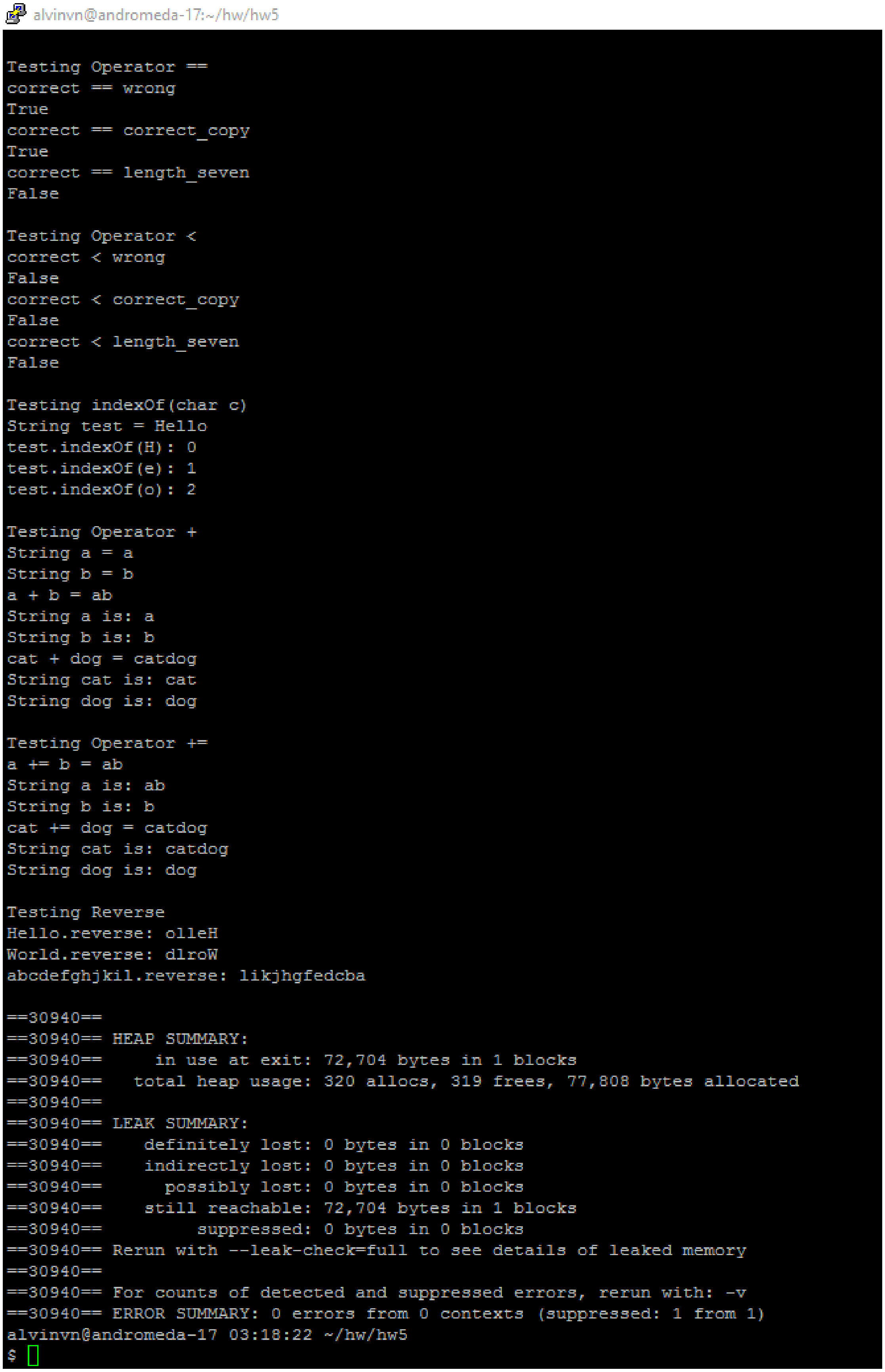
```
test.indexOf(e): 1
test.indexOf(o): 2

Testing Operator +
String a = a
String b = b
a + b = ab
String a is: a
String b is: b
cat + dog = catdog
String cat is: cat
String dog is: dog

Testing Operator +=
a += b = ab
String a is: ab
String b is: b
cat += dog = catdog
String cat is: catdog
String dog is: dog

Testing Reverse
Hello.reverse: olleH
World.reverse: dlroW
abcdefghjkil.reverse: likjhgfedcba

alvinvn@andromeda-17 03:14:25 ~/hw/hw5
```

VALGRIND FOR TEST_MAIN



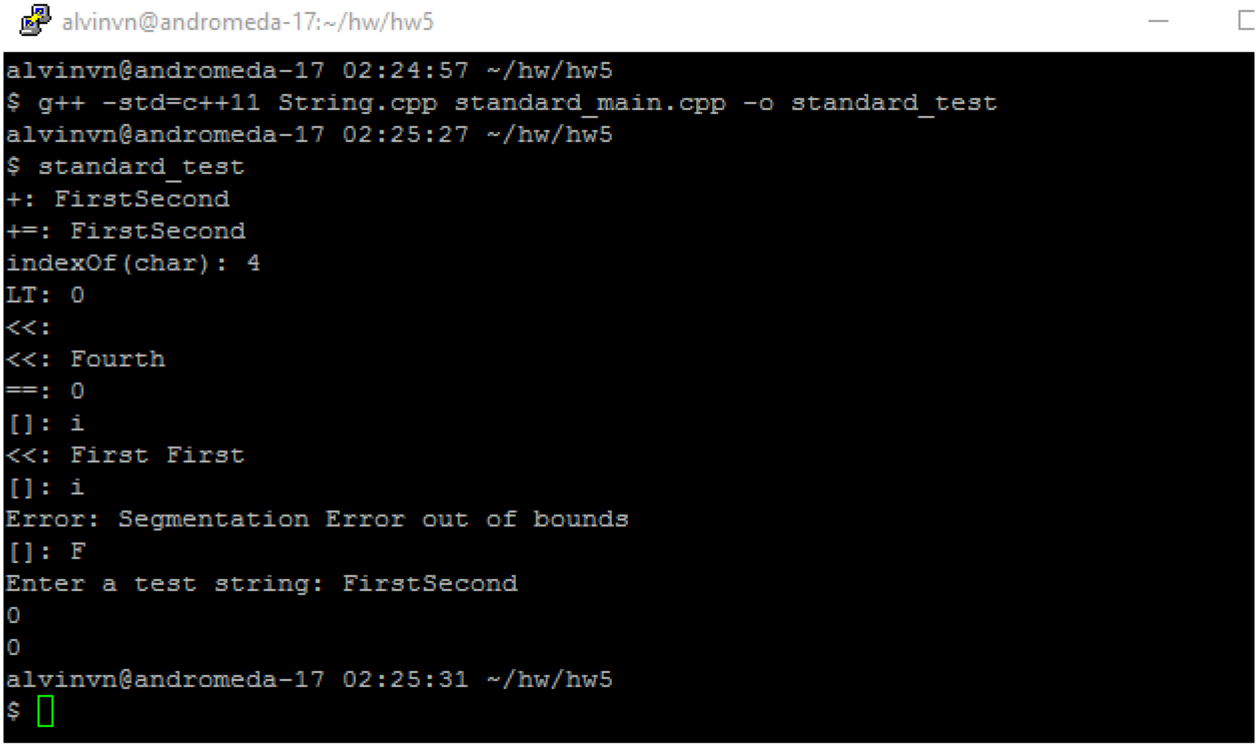
alvinvn@andromeda-17:~/hw/hw5

```
alvinvn@andromeda-17 03:18:15 ~/hw/hw5
$ g++ -std=c++11 String.cpp test_main.cpp -o test_main
alvinvn@andromeda-17 03:18:18 ~/hw/hw5
$ valgrind test_main
==30940== Memcheck, a memory error detector
==30940== Copyright (C) 2002-2013, and GNU GPL'd, by Julian Seward et al.
==30940== Using Valgrind-3.10.0 and LibVEX; rerun with -h for copyright info
==30940== Command: test_main
==30940==
Testing Constructor and Print
String a(Hello)
String b(Test1)
String c(Test2)
Hello
Test1
Test2

Testing Assignment Operator =
String test
String test1
String test2
test = Assigned
test1 = Assigned twice
test2 = Assigned three times
Assigned
Assigned twice
Assigned three times

Testing Operator Index []
String test = Hello
test[0]: H
test[1]: e
test[4]: o

Testing Size
String test = abcdefg
7
String test1 = weflkwjeafl
11
String test2 = werwlafhlasehflawfh
19

Testing Boolean Functions

String correct = Hello
String wrong = Helllo
String correct_copy = Hello
String length_seven = abcdefgh

Testing Operator ==
correct == wrong
True
correct == correct_copy
True
correct == length_seven
False

Testing Operator <
correct < wrong
False
correct < correct_copy
False
correct < length_seven
```



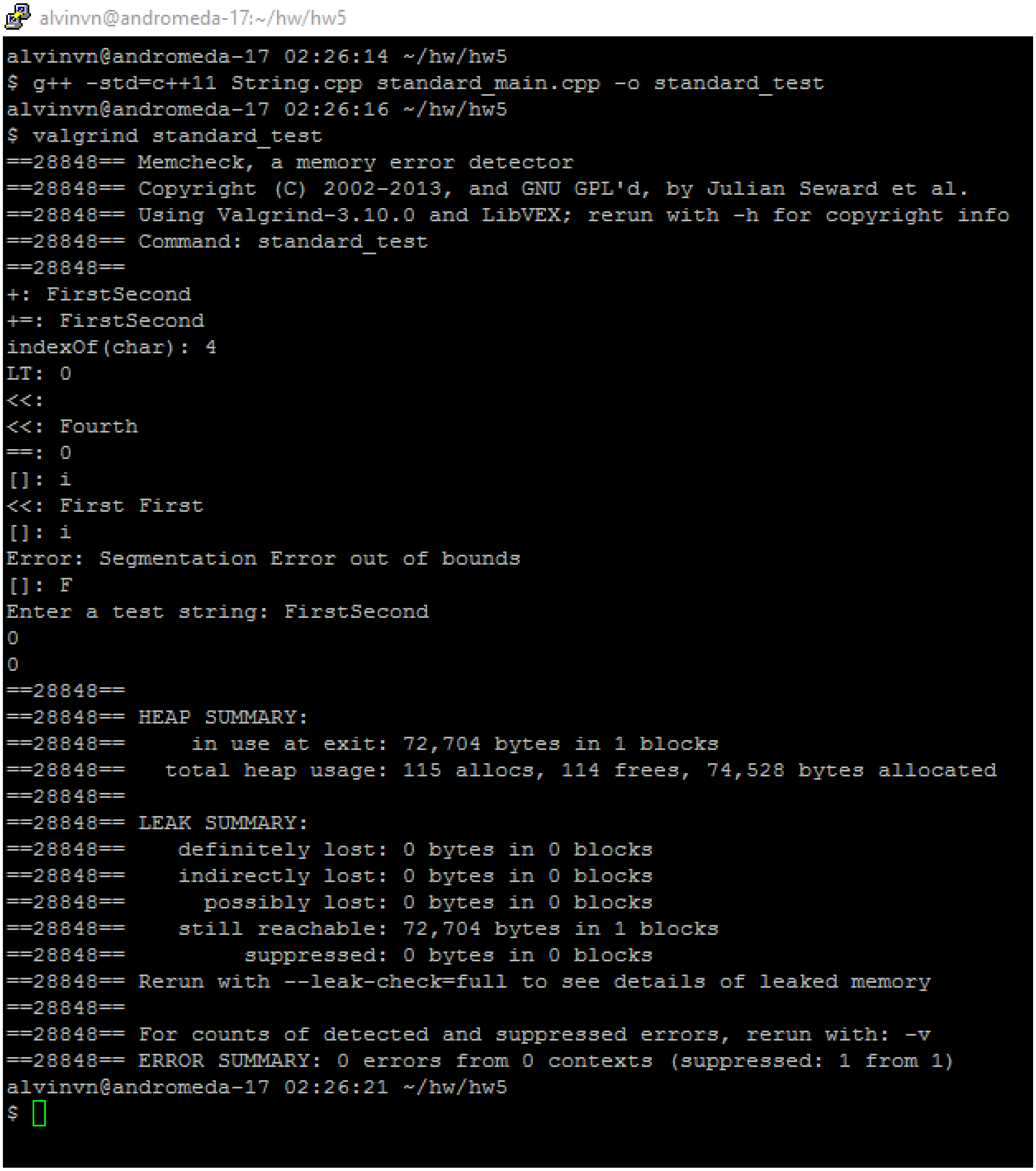
```
Testing Operator ==
correct == wrong
True
correct == correct_copy
True
correct == length_seven
False

Testing Operator <
correct < wrong
False
correct < correct_copy
False
correct < length_seven
False

Testing indexOf(char c)
String test = Hello
test.indexOf(H): 0
test.indexOf(e): 1
test.indexOf(o): 2

Testing Operator +
String a = a
String b = b
a + b = ab
String a is: a
String b is: b
cat + dog = catdog
String cat is: cat
String dog is: dog

Testing Operator +=
a += b = ab
String a is: ab
String b is: b
cat += dog = catdog
String cat is: catdog
String dog is: dog

Testing Reverse
Hello.reverse: olleH
World.reverse: dlroW
abcdefghjkil.reverse: likjhgfedcba

==30940==
==30940== HEAP SUMMARY:
==30940==     in use at exit: 72,704 bytes in 1 blocks
==30940==   total heap usage: 320 allocs, 319 frees, 77,808 bytes allocated
==30940==
==30940== LEAK SUMMARY:
==30940==    definitely lost: 0 bytes in 0 blocks
==30940==    indirectly lost: 0 bytes in 0 blocks
==30940==      possibly lost: 0 bytes in 0 blocks
==30940==    still reachable: 72,704 bytes in 1 blocks
==30940==         suppressed: 0 bytes in 0 blocks
==30940== Rerun with --leak-check=full to see details of leaked memory
==30940==
==30940== For counts of detected and suppressed errors, rerun with: -v
==30940== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 1 from 1)
alvinvn@andromeda-17 03:18:22 ~/hw/hw5
$ 
```

STANDARD_MAIN



```
alvinvn@andromeda-17 02:24:57 ~/hw/hw5
$ g++ -std=c++11 String.cpp standard_main.cpp -o standard_test
alvinvn@andromeda-17 02:25:27 ~/hw/hw5
$ standard_test
+: FirstSecond
+=: FirstSecond
indexOf(char): 4
LT: 0
<<:
<<: Fourth
==: 0
[]: i
<<: First First
[]: i
Error: Segmentation Error out of bounds
[]: F
Enter a test string: FirstSecond
0
0
alvinvn@andromeda-17 02:25:31 ~/hw/hw5
$ 
```

VALGRIND FOR STANDARD_MAIN



```
alvinvn@andromeda-17 02:26:14 ~/hw/hw5
$ g++ -std=c++11 String.cpp standard_main.cpp -o standard_test
alvinvn@andromeda-17 02:26:16 ~/hw/hw5
$ valgrind standard_test
==28848== Memcheck, a memory error detector
==28848== Copyright (C) 2002-2013, and GNU GPL'd, by Julian Seward et al.
==28848== Using Valgrind-3.10.0 and LibVEX; rerun with -h for copyright info
==28848== Command: standard_test
==28848==
+: FirstSecond
+=: FirstSecond
indexOf(char): 4
LT: 0
<<:
<<: Fourth
==: 0
[]: i
<<: First First
[]: i
Error: Segmentation Error out of bounds
[]: F
Enter a test string: FirstSecond
0
0
==28848==
==28848== HEAP SUMMARY:
==28848==     in use at exit: 72,704 bytes in 1 blocks
==28848==   total heap usage: 115 allocs, 114 frees, 74,528 bytes allocated
==28848==
==28848== LEAK SUMMARY:
==28848==    definitely lost: 0 bytes in 0 blocks
==28848==    indirectly lost: 0 bytes in 0 blocks
==28848==      possibly lost: 0 bytes in 0 blocks
==28848==    still reachable: 72,704 bytes in 1 blocks
==28848==         suppressed: 0 bytes in 0 blocks
==28848== Rerun with --leak-check=full to see details of leaked memory
==28848==
==28848== For counts of detected and suppressed errors, rerun with: -v
==28848== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 1 from 1)
alvinvn@andromeda-17 02:26:21 ~/hw/hw5
$
```