Name: Jonathan Smoley

Assignment: 3

Complete & Correct:	42 / 42
Tests:	6 / 6
Format and Comments:	6 / 6
Writeup:	6 / 6
Total Score:	60 / 60

Comments:

- 1. copy constructor can just call assignment operator (so you don't have duplicate code)
- 2. don't need to always use this->, it is distracting and unecessary

Unit test output:

```
==63774== Memcheck, a memory error detector
==63774== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==63774== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==63774== Command: ./hw3_test
==63774==
[======] Running 20 tests from 1 test suite.
[-----] Global test environment set-up.
[-----] 20 tests from BasicArraySeqTests
           ] BasicArraySeqTests.EmptySeqSize
[ RUN
        OK ] BasicArraySeqTests.EmptySeqSize (11 ms)
Γ
[ RUN
           ] BasicArraySeqTests.EmptySeqContains
        OK ] BasicArraySeqTests.EmptySeqContains (1 ms)
Γ
[ RUN
           ] BasicArraySeqTests.EmptySeqMemberAccess
Γ
        OK ] BasicArraySeqTests.EmptySeqMemberAccess (28 ms)
           ] BasicArraySeqTests.AddAndCheckSize
[ RUN
Γ
        OK ] BasicArraySeqTests.AddAndCheckSize (7 ms)
[ RUN
           ] BasicArraySeqTests.AddAndCheckContains
        OK ] BasicArraySeqTests.AddAndCheckContains (5 ms)
Γ
           ] BasicArraySeqTests.OutOfBoundsInsertIndexes
[ RUN
        OK ] BasicArraySeqTests.OutOfBoundsInsertIndexes (4 ms)
[ RUN
           ] BasicArraySeqTests.EraseAndCheckSize
        OK ] BasicArraySeqTests.EraseAndCheckSize (11 ms)
Γ
[ RUN
           ] BasicArraySeqTests.EraseAndCheckContains
        OK ] BasicArraySeqTests.EraseAndCheckContains (8 ms)
Γ
[ RUN
           ] BasicArraySeqTests.OutOfBoundsEraseIndexes
Γ
        OK ] BasicArraySeqTests.OutOfBoundsEraseIndexes (4 ms)
[ RUN
           ] BasicArraySeqTests.DestructorNoThrowChecksWithNew
Ε
        OK ] BasicArraySeqTests.DestructorNoThrowChecksWithNew (2 ms)
[ RUN
           ] BasicArraySeqTests.CopyConstructorChecks
Γ
        OK ] BasicArraySeqTests.CopyConstructorChecks (6 ms)
[ RUN
           ] BasicArraySeqTests.MoveConstructorChecks
        OK ] BasicArraySeqTests.MoveConstructorChecks (6 ms)
[ RUN
           ] BasicArraySeqTests.CopyAssignmentOpChecks
          ] BasicArraySeqTests.CopyAssignmentOpChecks (9 ms)
           ] BasicArraySeqTests.CheckRValueAccess
[ RUN
        OK ] BasicArraySeqTests.CheckRValueAccess (2 ms)
Γ
[ RUN
           ] BasicArraySeqTests.CheckLValueAccess
```

```
OK ] BasicArraySeqTests.CheckLValueAccess (3 ms)
[ RUN
          BasicArraySeqTests.OutOfBoundsLValueAccess
Γ
       OK ] BasicArraySeqTests.OutOfBoundsLValueAccess (3 ms)
[ RUN
          ] BasicArraySeqTests.OutOfBoundsRValueAccess
Ε
       OK ] BasicArraySeqTests.OutOfBoundsRValueAccess (2 ms)
[ RUN
          ] BasicArraySeqTests.StringInsertionChecks
       OK ] BasicArraySeqTests.StringInsertionChecks (10 ms)
Γ
[ RUN
          ] BasicArraySeqTests.CheckClear
[
       OK ] BasicArraySeqTests.CheckClear (2 ms)
[ RUN
          ] BasicArraySeqTests.ResizeCheck
        OK ] BasicArraySeqTests.ResizeCheck (3 ms)
[-----] 20 tests from BasicArraySeqTests (147 ms total)
[-----] Global test environment tear-down
[======] 20 tests from 1 test suite ran. (187 ms total)
[ PASSED ] 20 tests.
==63774==
==63774== HEAP SUMMARY:
           in use at exit: 0 bytes in 0 blocks
==63774== total heap usage: 677 allocs, 677 frees, 157,601 bytes allocated
==63774==
==63774== All heap blocks were freed -- no leaks are possible
==63774==
==63774== For lists of detected and suppressed errors, rerun with: -s
==63774== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

Performance test output:

```
# All times in milliseconds (msec)
# Column 1 = input data size
# Column 2 = avg time arrayseq insert front
# Column 3 = avg time cpp-vector insert front
# Column 4 = avg time arrayseq insert middle
# Column 5 = avg time cpp-vector insert middle
# Column 6 = avg time arrayseq insert end
\# Column 7 = avg time cpp-vector insert end
# Column 8 = avg time arrayseq update front
# Column 9 = avg time cpp-vector update front
# Column 10 = avg time arrayseq update middle
# Column 11 = avg time cpp-vector update middle
# Column 12 = avg time arrayseq update end
# Column 13 = avg time cpp-vector update end
# Column 14 = avg time arrayseq erase front
# Column 15 = avg time cpp-vector erase front
# Column 16 = avg time arrayseq erase middle
# Column 17 = avg time cpp-vector erase middle
# Column 18 = avg time arrayseg erase end
# Column 19 = avg time cpp-vector erase end
```

 $8388608 \ 29.60 \ 5.40 \ 11.20 \ 1.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 24.20 \ 2.20 \ 12.00 \ 1.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 26.20 \ 3.00 \ 13.00 \ 1.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 26.20 \ 3.00 \ 13.00 \ 1.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 29.00 \ 3.00 \ 14.20 \ 1.20 \ 0.00 \ 0.00 \ 0.00 \ 10000000 \ 30.40 \ 3.20 \ 15.20 \ 1.40 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 32.00 \ 3.20 \ 16.00 \ 1.40 \ 0.00 \ 0.00 \ 12000000 \ 33.40 \ 4.00 \ 16.60 \ 2.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 0.00 \ 35.00 \ 4.00 \ 17.20 \ 1.60 \ 0.00 \$