Name: *Jonathan Smoley*

Assignment: 5

Complete & Correct:	....	48 / 48
Tests:	....	0 / 0
Format and Comments:	....	6 / 6
Writeup:	....	6 / 6
Total Score:	....	60 / 60

Comments:

1. next and prev key should not be calling sort (not efficient), and instead each can be done in $O(n)$ time

Unit test output:

```
==325519== Memcheck, a memory error detector
==325519== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==325519== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==325519== Command: ./hw5_test
==325519==
[==========] Running 22 tests from 2 test suites.
[----------] Global test environment set-up.
[----------] 11 tests from BasicArrayMapTests
[ RUN      ] BasicArrayMapTests.EmptyCheck
[       OK ] BasicArrayMapTests.EmptyCheck (12 ms)
[ RUN      ] BasicArrayMapTests.InsertCheck
[       OK ] BasicArrayMapTests.InsertCheck (4 ms)
[ RUN      ] BasicArrayMapTests.RValueAccessCheck
[       OK ] BasicArrayMapTests.RValueAccessCheck (4 ms)
[ RUN      ] BasicArrayMapTests.LValueAccessCheck
[       OK ] BasicArrayMapTests.LValueAccessCheck (3 ms)
[ RUN      ] BasicArrayMapTests.ContainsCheck
[       OK ] BasicArrayMapTests.ContainsCheck (3 ms)
[ RUN      ] BasicArrayMapTests.EraseCheck
[       OK ] BasicArrayMapTests.EraseCheck (6 ms)
[ RUN      ] BasicArrayMapTests.KeyRangeCheck
[       OK ] BasicArrayMapTests.KeyRangeCheck (8 ms)
[ RUN      ] BasicArrayMapTests.SortedKeyCheck
[       OK ] BasicArrayMapTests.SortedKeyCheck (9 ms)
[ RUN      ] BasicArrayMapTests.InvalidKeyCheck
[       OK ] BasicArrayMapTests.InvalidKeyCheck (30 ms)
[ RUN      ] BasicArrayMapTests.NextKeyCheck
[       OK ] BasicArrayMapTests.NextKeyCheck (7 ms)
[ RUN      ] BasicArrayMapTests.PrevKeyCheck
[       OK ] BasicArrayMapTests.PrevKeyCheck (7 ms)
[----------] 11 tests from BasicArrayMapTests (106 ms total)

[----------] 11 tests from BasicLinkedMapTests
[ RUN      ] BasicLinkedMapTests.EmptyCheck
[       OK ] BasicLinkedMapTests.EmptyCheck (2 ms)
[ RUN      ] BasicLinkedMapTests.InsertCheck
[       OK ] BasicLinkedMapTests.InsertCheck (3 ms)
```

```
[ RUN      ] BasicLinkedMapTests.RValueAccessCheck
[       OK ] BasicLinkedMapTests.RValueAccessCheck (4 ms)
[ RUN      ] BasicLinkedMapTests.LValueAccessCheck
[       OK ] BasicLinkedMapTests.LValueAccessCheck (3 ms)
[ RUN      ] BasicLinkedMapTests.ContainsCheck
[       OK ] BasicLinkedMapTests.ContainsCheck (4 ms)
[ RUN      ] BasicLinkedMapTests.EraseCheck
[       OK ] BasicLinkedMapTests.EraseCheck (6 ms)
[ RUN      ] BasicLinkedMapTests.KeyRangeCheck
[       OK ] BasicLinkedMapTests.KeyRangeCheck (6 ms)
[ RUN      ] BasicLinkedMapTests.SortedKeyCheck
[       OK ] BasicLinkedMapTests.SortedKeyCheck (5 ms)
[ RUN      ] BasicLinkedMapTests.InvalidKeyCheck
[       OK ] BasicLinkedMapTests.InvalidKeyCheck (5 ms)
[ RUN      ] BasicLinkedMapTests.NextKeyCheck
[       OK ] BasicLinkedMapTests.NextKeyCheck (7 ms)
[ RUN      ] BasicLinkedMapTests.PrevKeyCheck
[       OK ] BasicLinkedMapTests.PrevKeyCheck (7 ms)
[----------] 11 tests from BasicLinkedMapTests (60 ms total)

[----------] Global test environment tear-down
[==========] 22 tests from 2 test suites ran. (206 ms total)
[  PASSED  ] 22 tests.
==325519==
==325519== HEAP SUMMARY:
==325519==     in use at exit: 0 bytes in 0 blocks
==325519==   total heap usage: 814 allocs, 814 frees, 153,006 bytes allocated
==325519==
==325519== All heap blocks were freed -- no leaks are possible
==325519==
==325519== 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 = array map insert
# Column 3 = linked map insert
# Column 4 = array map erase
# Column 5 = linked map erase
# Column 6 = array map contains
# Column 7 = linked map contains
# Column 8 = array map find range
# Column 9 = linked map find range
# Column 10 = array map next key
# Column 11 = linked map next key
# Column 12 = array map sorted keys
# Column 13 = linked map sorted keys
0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
2000 0.00 0.00 0.04 20.23 0.01 5.70 0.01 8.83 0.21 4.88 0.18 4.86
4000 0.00 0.00 0.03 38.01 0.02 19.01 0.03 35.74 0.45 19.38 0.41 19.49
6000 0.00 0.00 0.05 86.07 0.02 42.79 0.04 79.42 0.67 43.34 0.63 43.36
8000 0.00 0.00 0.07 154.08 0.03 75.97 0.05 132.65 0.91 76.65 0.85 77.77
10000 0.00 0.00 0.08 238.71 0.04 118.96 0.07 208.29 1.16 119.56 1.11 119.92
12000 0.00 0.00 0.10 346.01 0.11 171.91 0.08 296.52 1.41 172.45 1.33 172.25
14000 0.00 0.00 0.12 469.19 0.06 232.75 0.08 377.27 1.64 233.89 1.64 234.05
16000 0.00 0.00 0.13 615.17 0.07 303.89 0.10 497.11 1.89 304.77 1.84 305.56
18000 0.00 0.00 0.15 791.99 0.08 385.75 0.11 617.45 2.17 386.67 2.09 386.33
20000 0.00 0.00 0.16 969.28 0.08 475.72 0.12 715.76 2.51 495.36 2.43 482.84
36.78user 0.00system 0:36.80elapsed 99%CPU (0avgtext+0avgdata 4888maxresident)k
0inputs+64outputs (0major+586minor)pagefaults 0swaps
```