```
// $Id: vectorshared.cpp,v 1.4 2014-04-22 19:02:44-07 - - $

//
// Exact copy of vectorleak.cpp, except use shared_ptr instead of
// actual pointers so that memory is released.
//

#include <iostream>
#include <vector>
#include <memory>

using namespace std;

int main (int argc, char **argv) {
   vector<shared_ptr<string>> vs;
   for (int index = 1; index < argc; ++index) {
      vs.push_back (make_shared<string> (argv[index]));
   }
   auto begin = vs.begin();
   for (auto itor = begin; itor != vs.end(); ++itor) {
      cout << itor - begin << ": " << *itor << "->" << **itor << endl;
   }
   return 0;
}

/*
//TEST// alias grind='valgrind --leak-check=full --show-reachable=yes'
//TEST// grind vectorshared these are some arguments to check on leak \
//TEST//       >vectorshared.out 2>&1
//TEST// mkpspdf vectorshared.ps vectorshared.cpp* vectorshared.out
*/

```

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: starting vectorshared.cpp
vectorshared.cpp: $Id: vectorshared.cpp,v 1.4 2014-04-22 19:02:44-07 - - $
g++ -g -O0 -Wall -Wextra -std=gnu++11 vectorshared.cpp -o vectorshared -lm
rm -f vectorshared.o
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: finished vectorshared.cpp
```

```
==4198== Memcheck, a memory error detector
==4198== Copyright (C) 2002-2012, and GNU GPL'd, by Julian Seward et al.
==4198== Using Valgrind-3.8.1 and LibVEX; rerun with -h for copyright info
==4198== Command: vectorshared these are some arguments to check on leak
==4198==
0: 0x4c2b058->these
1: 0x4c2b168->are
2: 0x4c2b288->some
3: 0x4c2b3c8->arguments
4: 0x4c2b498->to
5: 0x4c2b618->check
6: 0x4c2b6d8->on
7: 0x4c2b798->leak
==4198==
==4198== HEAP SUMMARY:
==4198==     in use at exit: 0 bytes in 0 blocks
==4198==   total heap usage: 20 allocs, 20 frees, 730 bytes allocated
==4198==
==4198== All heap blocks were freed -- no leaks are possible
==4198==
==4198== For counts of detected and suppressed errors, rerun with: -v
==4198== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 6 from 6)
```