```
 1: // $Id: linkstrstack.cpp,v 1.4 2013-08-08 14:45:56-07 - - $
 2:
 3: //
 4: // linkstrstack - show the linked list implementation of a stackk
 5: //
 6:
 7: #include <cstddef>
 8: #include <iostream>
 9: #include <stdexcept>
10: #include <string>
11:
12: using namespace std;
13:
14: //////////////////////////////////////////////////////////////////
15: // linkstrstack.h
16: //////////////////////////////////////////////////////////////////
17:
18: class linkstrstack {
19:    private:
20:       struct node {
21:          string item;
22:          node *link;
23:       };
24:       node *topnode;
25:       int count;
26:       linkstrstack (const linkstrstack &) = delete;
27:       linkstrstack (linkstrstack &&) = delete;
28:       linkstrstack &operator= (const linkstrstack &) = delete;
29:       linkstrstack &operator= (linkstrstack &&) = delete;
30:    public:
31:       linkstrstack();
32:       ~linkstrstack();
33:       void push (const string &);
34:       void pop();
35:       string &top();
36:       const string &top() const;
37:       size_t size() const;
38:       bool empty() const;
39: };
40:
```

```
41:
42: //////////////////////////////////////////////////////////////////
43: // linkstrstack.cpp
44: //////////////////////////////////////////////////////////////////
45:
46: linkstrstack::linkstrstack(): topnode (NULL), count(0) {
47: }
48:
49: linkstrstack::~linkstrstack() {
50:    while (not empty()) pop();
51: }
52:
53: void linkstrstack::push (const string &item) {
54:    node *tmp = new node();
55:    tmp->item = item;
56:    tmp->link = topnode;
57:    topnode = tmp;
58:    ++count;
59: }
60:
61: void linkstrstack::pop() {
62:    if (empty()) throw out_of_range ("linkstrstack::pop()");
63:    node *tmp = topnode;
64:    topnode = topnode->link;
65:    delete tmp;
66:    --count;
67: }
68:
69: string &linkstrstack::top() {
70:    if (empty()) throw out_of_range ("linkstrstack::top()");
71:    return topnode->item;
72: }
73:
74: const string &linkstrstack::top() const {
75:    if (empty()) throw out_of_range ("linkstrstack::top()");
76:    return topnode->item;
77: }
78:
79: size_t linkstrstack::size() const {
80:    return count;
81: }
82:
83: bool linkstrstack::empty() const {
84:    return count == 0;
85: }
86:
```

```
 87:
 88: //////////////////////////////////////////////////////////////////////
 89: // main.cpp
 90: //////////////////////////////////////////////////////////////////////
 91:
 92: int main (int argc, char **argv) {
 93:
 94:    linkstrstack stkstr;
 95:    for (int argi = 1; argi < argc; ++argi) {
 96:       stkstr.push (argv[argi]);
 97:    }
 98:    while (stkstr.size() > size_t (argc / 2)) {
 99:       cout << stkstr.top() << endl;
100:       stkstr.pop();
101:    }
102:
103:    return 0;
104: }
105:
106: /*
107: //TEST// valgrind --leak-check=full --show-reachable=yes \
108: //TEST//       --log-file=linkstrstack.out.grind \
109: //TEST//       linkstrstack this is some test data for the stack \
110: //TEST//       >linkstrstack.out 2>&1
111: //TEST// mkpspdf linkstrstack.ps linkstrstack.cpp* linkstrstack.out*
112: */
113:
```

```
    1: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: starting linkstrstack.cpp
    2: linkstrstack.cpp: $Id: linkstrstack.cpp,v 1.4 2013-08-08 14:45:56-07 - -
 $
    3: g++ -g -O0 -Wall -Wextra -std=gnu++11 linkstrstack.cpp -o linkstrstack -
lm
    4: rm -f linkstrstack.o
    5: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: finished linkstrstack.cpp
```

```
1: stack
2: the
3: for
4: data
```

```
    1: ==10275== Memcheck, a memory error detector
    2: ==10275== Copyright (C) 2002-2012, and GNU GPL'd, by Julian Seward et al
.
    3: ==10275== Using Valgrind-3.8.1 and LibVEX; rerun with -h for copyright i
nfo
    4: ==10275== Command: linkstrstack this is some test data for the stack
    5: ==10275== Parent PID: 10274
    6: ==10275==
    7: ==10275==
    8: ==10275== HEAP SUMMARY:
    9: ==10275==     in use at exit: 0 bytes in 0 blocks
   10: ==10275==   total heap usage: 16 allocs, 16 frees, 357 bytes allocated
   11: ==10275==
   12: ==10275== All heap blocks were freed -- no leaks are possible
   13: ==10275==
   14: ==10275== For counts of detected and suppressed errors, rerun with: -v
   15: ==10275== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 6 from 6)
```