```
 1: // $Id: queue_vec.cpp,v 1.10 2014-07-18 15:53:49-07 - - $
 2:
 3: #include <algorithm>
 4: #include <iostream>
 5: #include <vector>
 6: using namespace std;
 7:
 8: template <typename item_t>
 9: class queue {
10:    private:
11:       vector<item_t> front_vec;
12:       vector<item_t> back_vec;
13:       size_t front_pos {};
14:       void maybe_switch();
15:    public:
16:       item_t& front() { return front_vec[front_pos]; }
17:       const item_t& front() const { return front_vec[front_pos]; }
18:       void pop_front();
19:       void push_back (const item_t& val);
20:       void push_back (item_t&& val);
21:       size_t size() const;
22:       bool empty() const { return size() == 0; }
23: };
24:
25: template <typename item_t>
26: void queue<item_t>::maybe_switch() {
27:    if (front_pos == front_vec.size()) {
28:       front_vec.clear();
29:       front_pos = 0;
30:       if (back_vec.size() > 0) swap (front_vec, back_vec);
31:    }
32: }
33:
34: template <typename item_t>
35: void queue<item_t>::pop_front() {
36:    item_t tmp = std::move (front_vec[front_pos++]);
37:    maybe_switch();
38: }
39:
40: template <typename item_t>
41: void queue<item_t>::push_back (const item_t& val) {
42:    back_vec.push_back (val);
43:    maybe_switch();
44: }
45:
46: template <typename item_t>
47: void queue<item_t>::push_back (item_t&& val) {
48:    back_vec.push_back (val);
49:    maybe_switch();
50: }
51:
52: template <typename item_t>
53: size_t queue<item_t>::size() const {
54:    return back_vec.size() + front_vec.size() - front_pos;
55: }
56:
```

```
57:
58: int main (int argc, char** argv) {
59:    queue<string> que;
60:    for_each (&argv[0], &argv[argc],
61:              [&] (char* arg) { que.push_back (arg); });
62:    while (not que.empty()) {
63:       cout << que.front() << endl;
64:       que.pop_front();
65:    }
66:    cout << "sizeof queue<string> = " << sizeof (queue<string>) << endl;
67: }
68:
69: /*
70: //TEST// alias grind='valgrind --leak-check=full --show-reachable=yes'
71: //TEST// grind queue_vec This is a test of queue_vec. \
72: //TEST//       >queue_vec.out 2>&1
73: //TEST// mkpspdf queue_vec.ps queue_vec.cpp* queue_vec.out
74: */
75:
```

```
    1: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: starting queue_vec.cpp
    2: queue_vec.cpp:
    3:       $Id: queue_vec.cpp,v 1.10 2014-07-18 15:53:49-07 - - $
    4: g++ -g -O0 -Wall -Wextra -rdynamic -std=gnu++11 queue_vec.cpp -o queue_v
ec -lglut -lGLU -lGL -lX11 -lrt -lm
    5: rm -f queue_vec.o
    6: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: finished queue_vec.cpp
```

```
    1: ==21303== Memcheck, a memory error detector
    2: ==21303== Copyright (C) 2002-2012, and GNU GPL'd, by Julian Seward et al
.
    3: ==21303== Using Valgrind-3.8.1 and LibVEX; rerun with -h for copyright i
nfo
    4: ==21303== Command: queue_vec This is a test of queue_vec.
    5: ==21303==
    6: queue_vec
    7: This
    8: is
    9: a
   10: test
   11: of
   12: queue_vec.
   13: sizeof queue<string> = 56
   14: ==21303==
   15: ==21303== HEAP SUMMARY:
   16: ==21303==     in use at exit: 0 bytes in 0 blocks
   17: ==21303==   total heap usage: 13 allocs, 13 frees, 344 bytes allocated
   18: ==21303==
   19: ==21303== All heap blocks were freed -- no leaks are possible
   20: ==21303==
   21: ==21303== For counts of detected and suppressed errors, rerun with: -v
   22: ==21303== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 6 from 6)
```