```
 1: // $Id: listfree.cpp,v 1.24 2019-01-10 13:58:49-08 - - $
 2:
 3: // Show how to break a cycle in a simple circular list.
 4:
 5: #include <algorithm>
 6: #include <iostream>
 7: #include <memory>
 8: using namespace std;
 9:
10: struct node;
11:
12: using node_ptr = shared_ptr<node>;
13:
14: struct node {
15:    int value;
16:    node_ptr link;
17:    node (int value_, node_ptr link_): value(value_), link(link_) {}
18: };
19:
20: int main (int argc, char** argv) {
21:    cout << "Command:";
22:    for_each (&argv[0], &argv[argc], [](char* arg){cout << " " << arg;});
23:    cout << endl;
24:    bool break_cycle = argc > 1 and argv[1] == "-f"s;
25:    node_ptr list = make_shared<node> (1,
26:                    make_shared<node> (2,
27:                    make_shared<node> (3, nullptr)));
28:    list->link->link->link = list;
29:    cout << "list = " << list << endl;
30:    for (auto curr = list;;) {
31:       cout << curr << " -> {" << curr->value << ", " << curr->link
32:            << "} (use_count " << curr.use_count() << ")" << endl;
33:       curr = curr->link;
34:       if (curr == list) break;
35:    }
36:    if (break_cycle) list->link = nullptr;
37:    return 0;
38: }
39:
40: //TEST// valgrind listfree -0 >listfree.out-0 2>&1
41: //TEST// valgrind listfree -f >listfree.out-f 2>&1
42: //TEST// mkpspdf listfree.ps listfree.cpp* listfree.out-*
43:
```

```
 1: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: starting listfree.cpp
 2: checksource listfree.cpp
 3: ident listfree.cpp
 4: listfree.cpp:
 5:      $Id: listfree.cpp,v 1.24 2019-01-10 13:58:49-08 - - $
 6: cpplint.py.perl listfree.cpp
 7: Done processing listfree.cpp
 8: g++ -Wall -Wextra -Wpedantic -Wshadow -fdiagnostics-color=never -std=gnu
++2a -Wold-style-cast -g -O0 listfree.cpp -o listfree -lm
 9: rm -f listfree.o
10: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: finished listfree.cpp
```

```
 1: ==22235== Memcheck, a memory error detector
 2: ==22235== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al
.
 3: ==22235== Using Valgrind-3.14.0 and LibVEX; rerun with -h for copyright
info
 4: ==22235== Command: listfree -0
 5: ==22235==
 6: Command: listfree -0
 7: list = 0x5a24190
 8: 0x5a24190 -> {1, 0x5a24120} (use_count 3)
 9: 0x5a24120 -> {2, 0x5a240b0} (use_count 2)
10: 0x5a240b0 -> {3, 0x5a24190} (use_count 2)
11: ==22235==
12: ==22235== HEAP SUMMARY:
13: ==22235==     in use at exit: 120 bytes in 3 blocks
14: ==22235==   total heap usage: 4 allocs, 1 frees, 147 bytes allocated
15: ==22235==
16: ==22235== LEAK SUMMARY:
17: ==22235==    definitely lost: 40 bytes in 1 blocks
18: ==22235==    indirectly lost: 80 bytes in 2 blocks
19: ==22235==      possibly lost: 0 bytes in 0 blocks
20: ==22235==    still reachable: 0 bytes in 0 blocks
21: ==22235==         suppressed: 0 bytes in 0 blocks
22: ==22235== Rerun with --leak-check=full to see details of leaked memory
23: ==22235==
24: ==22235== For counts of detected and suppressed errors, rerun with: -v
25: ==22235== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

```
 1: ==22236== Memcheck, a memory error detector
 2: ==22236== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al
.
 3: ==22236== Using Valgrind-3.14.0 and LibVEX; rerun with -h for copyright
info
 4: ==22236== Command: listfree -f
 5: ==22236==
 6: Command: listfree -f
 7: list = 0x5a24190
 8: 0x5a24190 -> {1, 0x5a24120} (use_count 3)
 9: 0x5a24120 -> {2, 0x5a240b0} (use_count 2)
10: 0x5a240b0 -> {3, 0x5a24190} (use_count 2)
11: ==22236==
12: ==22236== HEAP SUMMARY:
13: ==22236==     in use at exit: 0 bytes in 0 blocks
14: ==22236==   total heap usage: 4 allocs, 4 frees, 147 bytes allocated
15: ==22236==
16: ==22236== All heap blocks were freed -- no leaks are possible
17: ==22236==
18: ==22236== For counts of detected and suppressed errors, rerun with: -v
19: ==22236== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```