```
 1: // $Id: vtablefns.cpp,v 1.26 2015-02-10 17:55:14-08 - - $
 2:
 3: // Illustrate inheritance and virtual function tables.
 4: // Print out the name of the function actually called.
 5:
 6: #include <iostream>
 7: #include <string>
 8: #include <typeinfo>
 9:
10: using namespace std;
11: #define TRACE(STMT) cout << "TRACE (" << #STMT << ")" << endl; STMT
12:
13: string next() {
14:    static string name = "(0)";
15:    ++name[1];
16:    return name;
17: }
18:
19: struct AAA {
20:    string aname;
21:    AAA (const AAA&) = delete;
22:    AAA& operator= (const AAA&) = delete;
23:    AAA(): aname(next()) { show ("AAA::AAA()" + aname) << endl; }
24:    virtual ~AAA() { show ("AAA::~AAA()" + aname) << endl; }
25:    virtual string f () const { return "AAA::f()" + aname; }
26:    virtual string g () const { return "AAA::g()" + aname; }
27:    ostream& show (const string& str) const;
28: };
29:
30: struct BBB: public AAA {
31:    string bname;
32:    BBB(): bname(next()) { show ("BBB::BBB()" + aname + bname) << endl; }
33:    virtual ~BBB() { show ("BBB::~BBB()" + aname + bname) << endl; }
34:    virtual string f () const { return "BBB::f()" + aname + bname; }
35: };
36:
37: struct CCC: public AAA {
38:    string cname;
39:    CCC(): cname(next()) { show ("CCC::CCC()" + aname + cname) << endl; }
40:    virtual ~CCC() { show ("CCC::~CCC()" + aname + cname) << endl; }
41:    virtual string g () const { return "CCC::g()" + aname + cname; }
42: };
43:
44: ostream& AAA::show (const string& str) const {
45:    cout << this << "->" << str << ": typeid = \""
46:         << typeid (*this).name () << "\""; return cout;
47:    return cout;
48: }
49:
50: void tester (const AAA& p) {
51:    p.show ("tester") << ": f = " << p.f()
52:         << ", g = " << p.g() << endl << "." << endl;
53: }
54:
55: int main() {
56:    TRACE (AAA a; tester (a);)
57:    TRACE (BBB b; tester (b);)
58:    TRACE (CCC c; tester (c);)
```

```
59:    TRACE (return 0;)
60: }
61:
62: //TEST// grind="valgrind --leak-check=full --show-reachable=yes"
63: //TEST// $grind vtablefns >vtablefns.out 2>&1
64: //TEST// mkpspdf vtablefns.ps vtablefns.cpp* vtablefns.out*
65:
```

```
    1: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: starting vtablefns.cpp
    2: vtablefns.cpp:
    3:      $Id: vtablefns.cpp,v 1.26 2015-02-10 17:55:14-08 - - $
    4: g++ -g -O0 -Wall -Wextra -rdynamic -std=gnu++11 vtablefns.cpp -o vtablef
ns -lglut -lGLU -lGL -lX11 -lrt -lm
    5: rm -f vtablefns.o
    6: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: finished vtablefns.cpp
```

```
    1: ==9054== Memcheck, a memory error detector
    2: ==9054== Copyright (C) 2002-2013, and GNU GPL'd, by Julian Seward et al.
    3: ==9054== Using Valgrind-3.9.0 and LibVEX; rerun with -h for copyright in
fo
    4: ==9054== Command: vtablefns
    5: ==9054==
    6: TRACE (AAA a; tester (a);)
    7: 0xffefff960->AAA::AAA()(1): typeid = "3AAA"
    8: 0xffefff960->tester: typeid = "3AAA": f = AAA::f()(1), g = AAA::g()(1)
    9: .
   10: TRACE (BBB b; tester (b);)
   11: 0xffefff940->AAA::AAA()(2): typeid = "3AAA"
   12: 0xffefff940->BBB::BBB()(2)(3): typeid = "3BBB"
   13: 0xffefff940->tester: typeid = "3BBB": f = BBB::f()(2)(3), g = AAA::g()(2
)
   14: .
   15: TRACE (CCC c; tester (c);)
   16: 0xffefff920->AAA::AAA()(4): typeid = "3AAA"
   17: 0xffefff920->CCC::CCC()(4)(5): typeid = "3CCC"
   18: 0xffefff920->tester: typeid = "3CCC": f = AAA::f()(4), g = CCC::g()(4)(5
)
   19: .
   20: TRACE (return 0;)
   21: 0xffefff920->CCC::~CCC()(4)(5): typeid = "3CCC"
   22: 0xffefff920->AAA::~AAA()(4): typeid = "3AAA"
   23: 0xffefff940->BBB::~BBB()(2)(3): typeid = "3BBB"
   24: 0xffefff940->AAA::~AAA()(2): typeid = "3AAA"
   25: 0xffefff960->AAA::~AAA()(1): typeid = "3AAA"
   26: ==9054==
   27: ==9054== HEAP SUMMARY:
   28: ==9054==     in use at exit: 0 bytes in 0 blocks
   29: ==9054==   total heap usage: 32 allocs, 32 frees, 1,173 bytes allocated
   30: ==9054==
   31: ==9054== All heap blocks were freed -- no leaks are possible
   32: ==9054==
   33: ==9054== For counts of detected and suppressed errors, rerun with: -v
   34: ==9054== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 6 from 6)
```