```
1: // $Id: virtual.cpp,v 1.14 2015-02-10 17:54:20-08 - - $
 3: #include <iostream>
 4: #include <typeinfo>
 6: using namespace std;
7: #define TRACE(STMT) cout << "TRACE (" << #STMT << ")" << endl; STMT
8:
9: class nonbase {
10:
       public:
11:
          nonbase() { cout << "nonbase()" << endl; }</pre>
12:
          ~nonbase() { cout << "~nonbase()" << endl; }</pre>
13:
          void show () { cout << typeid (*this).name() << endl; }</pre>
14: };
15:
16: class nonderived: public nonbase {
17:
       public:
          nonderived() { cout << "nonderived()" << endl; }</pre>
18:
19:
          ~nonderived() { cout << "~nonderived()" << endl; }</pre>
          void show () { cout << typeid (*this).name() << endl; }</pre>
20:
21: };
22:
23: class virtbase {
      public:
24:
25:
          // Inheritance classes must have virtual destructors.
26:
          virtbase() { cout << "virtbase()" << endl; }</pre>
27:
          virtual ~virtbase() { cout << "~virtbase() " << endl; }</pre>
28:
          virtual void show () { cout << typeid (*this).name() << endl; }</pre>
29: };
30:
31: class virtderived: public virtbase {
32:
       public:
33:
          virtderived() { cout << "virtderived()" << endl; }</pre>
          virtual ~virtderived() { cout << "~virtderived()" << endl; }</pre>
34:
35:
          virtual void show () { cout << typeid (*this).name() << endl; }</pre>
36: };
37:
38: int main () {
39:
       TRACE (nonbase *nb = new nonbase();)
40:
       TRACE (nb->show();)
41:
       TRACE (nonderived *nd = new nonderived ();)
       TRACE (nd->show();)
42:
       TRACE (delete nb;)
43:
44:
       TRACE (nb = nd;)
45:
       TRACE (nb->show();)
46:
       TRACE (nd = static_cast<nonderived*> (nb);)
47:
       TRACE (nd->show();)
48:
       TRACE (delete nd;)
49:
50:
       TRACE (virtbase *vb = new virtbase();)
       TRACE (vb->show();)
51:
52:
       TRACE (virtderived *vd = new virtderived ();)
53:
       TRACE (vd->show();)
       TRACE (delete vb;)
54:
55:
       TRACE (vb = vd;)
56:
       TRACE (vb->show();)
       TRACE (vd = dynamic_cast<virtderived*> (vb);)
57:
58:
       TRACE (vd->show();)
```

02/10/15 17:54:20

\$cmps109-wm/Examples/wk06c-inheritance/virtual.cpp

2/2

```
59: TRACE (delete vd;)
60: TRACE (return 0;)
61: }
62:
63: //TEST// grind="valgrind --leak-check=full --show-reachable=yes"
64: //TEST// $grind virtual 1>virtual.out 2>&1
65: //TEST// mkpspdf virtual.ps virtual.cpp* virtual.out*
66:
```

02/10/15 17:54:20

\$cmps109-wm/Examples/wk06c-inheritance/virtual.cpp.log

1/1

```
1: ==8927== Memcheck, a memory error detector
    2: ==8927== Copyright (C) 2002-2013, and GNU GPL'd, by Julian Seward et al.
    3: ==8927== Using Valgrind-3.9.0 and LibVEX; rerun with -h for copyright in
fo
    4: ==8927== Command: virtual
    5: ==8927==
    6: TRACE (nonbase *nb = new nonbase();)
    7: nonbase()
    8: TRACE (nb->show();)
    9: 7nonbase
   10: TRACE (nonderived *nd = new nonderived ();)
   11: nonbase()
   12: nonderived()
   13: TRACE (nd->show();)
   14: 10nonderived
   15: TRACE (delete nb;)
   16: ~nonbase()
   17: TRACE (nb = nd;)
   18: TRACE (nb->show();)
   19: 7nonbase
   20: TRACE (nd = static_cast<nonderived*> (nb);)
   21: TRACE (nd->show();)
   22: 10nonderived
   23: TRACE (delete nd;)
   24: ~nonderived()
   25: ~nonbase()
   26: TRACE (virtbase *vb = new virtbase();)
   27: virtbase()
   28: TRACE (vb->show();)
   29: 8virtbase
   30: TRACE (virtderived *vd = new virtderived ();)
   31: virtbase()
   32: virtderived()
   33: TRACE (vd->show();)
   34: 11virtderived
   35: TRACE (delete vb;)
   36: ~virtbase()
   37: TRACE (vb = vd;)
   38: TRACE (vb->show();)
   39: 11virtderived
   40: TRACE (vd = dynamic_cast<virtderived*> (vb);)
   41: TRACE (vd->show();)
   42: 11virtderived
   43: TRACE (delete vd;)
   44: ~virtderived()
   45: ~virtbase()
   46: TRACE (return 0;)
   47: ==8927==
   48: ==8927== HEAP SUMMARY:
   49: ==8927== in use at exit: 0 bytes in 0 blocks
   50: ==8927==
                total heap usage: 5 allocs, 5 frees, 27 bytes allocated
   51: ==8927==
   52: ==8927== All heap blocks were freed -- no leaks are possible
   53: ==8927==
   54: ==8927== For counts of detected and suppressed errors, rerun with: -v
   55: ==8927== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 6 from 6)
```