```
 1: // $Id: inheritance.cpp,v 1.29 2015-02-10 17:43:42-08 - - $
 2:
 3: //
 4: // Example using objects, with a base object and two derived objects.
 5: //
 6:
 7: #include <iostream>
 8: #include <typeinfo>
 9: #include <memory>
10: #include <vector>
11:
12: using namespace std;
13:
14: //////////////////////////////////////////////////////////////////
15: // class object
16: //////////////////////////////////////////////////////////////////
17:
18: class object {
19:    private:
20:       object (const object&) = delete;
21:       object& operator= (const object&) = delete;
22:       static unsigned next_id;;
23:    protected:
24:       const unsigned id;
25:       object(); // abstract class, so only derived can used ctor.
26:    public:
27:       virtual ~object(); // must be virtual
28:       virtual void print (ostream&) const;
29: };
30:
31: ostream& operator<< (ostream& out, const object& obj) {
32:    obj.print (out);
33:    return out;
34: }
35:
36: unsigned object::next_id = 0;
37:
38: object::object(): id(++next_id) {
39:    cout << "Create: " << *this << endl;
40: }
41:
42: object::~object() {
43:    cout << "Delete: " << *this << endl;
44: }
45:
46: void object::print (ostream& out) const {
47:    out << "[" << static_cast<const void *const> (this) << "->"
48:        << typeid(*this).name() << "] id=" << id << ": ";
49: }
50:
```

```
 51:
 52: //////////////////////////////////////////////////////////////////
 53: // class rectangle
 54: //////////////////////////////////////////////////////////////////
 55:
 56: class rectangle: public object {
 57:    private:
 58:       size_t width;
 59:       size_t height;
 60:    public:
 61:       rectangle (size_t width = 0, size_t height = 0);
 62:       virtual ~rectangle();
 63:       virtual void print (ostream&) const;
 64: };
 65:
 66: rectangle::rectangle (size_t width, size_t height):
 67:             width(width), height(height) {
 68:    cout << "Create: " << *this << endl;
 69: }
 70:
 71: rectangle::~rectangle() {
 72:    cout << "Delete: " << *this << endl;
 73: }
 74:
 75: void rectangle::print (ostream& out) const {
 76:    this->object::print (out);
 77:    out << "rectangle: width=" << width << ", height=" << height;
 78: }
 79:
 80: //////////////////////////////////////////////////////////////////
 81: // class circle
 82: //////////////////////////////////////////////////////////////////
 83:
 84: class circle: public object {
 85:    private:
 86:       size_t diameter;
 87:    public:
 88:       circle (size_t diameter = 0);
 89:       virtual ~circle();
 90:       virtual void print (ostream&) const;
 91: };
 92:
 93: circle::circle (size_t diameter): diameter(diameter) {
 94:    cout << "Create: " << *this << endl;
 95: }
 96:
 97: circle::~circle() {
 98:    cout << "Delete: " << *this << endl;
 99: }
100:
101: void circle::print (ostream& out) const {
102:    this->object::print (out);
103:    out << "circle: " << "diameter=" << diameter;
104: }
105:
106:
```

```
107:
108: //////////////////////////////////////////////////////////////////
109: // main
110: //////////////////////////////////////////////////////////////////
111:
112: int main() {
113:    cout << "sizeof (object) = " << sizeof (object) << endl;
114:    cout << "sizeof (rectangle) = " << sizeof (rectangle) << endl;
115:    cout << "sizeof (circle) = " << sizeof (circle) << endl;
116:
117:    vector<shared_ptr<object>> vec;
118:    // ERROR: v.push_back (new object());
119:    // ERROR: object o;
120:    vec.push_back (shared_ptr<object> (new circle ()));
121:    vec.push_back (shared_ptr<object> (new circle (10)));
122:    vec.push_back (shared_ptr<object> (new rectangle()));
123:    vec.push_back (shared_ptr<object> (new rectangle (5)));
124:    vec.push_back (shared_ptr<object> (new rectangle (8)));
125:    cout << endl;
126:
127:    for (const auto& ptr: vec) {
128:       cout << "Object: " << *ptr << endl;
129:    }
130:    cout << endl;
131:
132:    cout << "return 0" << endl;
133:    return 0;
134: }
135:
136: /*
137: //TEST// valgrind --leak-check=full --show-reachable=yes \
138: //TEST//       inheritance >inheritance.out 2>&1
139: //TEST// mkpspdf inheritance.ps inheritance.cpp* inheritance.out*
140: */
141:
```

```
    1: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: starting inheritance.cpp
    2: inheritance.cpp:
    3:      $Id: inheritance.cpp,v 1.29 2015-02-10 17:43:42-08 - - $
    4: g++ -g -O0 -Wall -Wextra -rdynamic -std=gnu++11 inheritance.cpp -o inher
itance -lglut -lGLU -lGL -lX11 -lrt -lm
    5: rm -f inheritance.o
    6: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: finished inheritance.cpp
```

```
    1: ==6348== Memcheck, a memory error detector
    2: ==6348== Copyright (C) 2002-2013, and GNU GPL'd, by Julian Seward et al.
    3: ==6348== Using Valgrind-3.9.0 and LibVEX; rerun with -h for copyright in
fo
    4: ==6348== Command: inheritance
    5: ==6348==
    6: sizeof (object) = 16
    7: sizeof (rectangle) = 32
    8: sizeof (circle) = 24
    9: Create: [0x4e7d090->6object] id=1:
   10: Create: [0x4e7d090->6circle] id=1: circle: diameter=0
   11: Create: [0x4e7d1a0->6object] id=2:
   12: Create: [0x4e7d1a0->6circle] id=2: circle: diameter=10
   13: Create: [0x4e7d2c0->6object] id=3:
   14: Create: [0x4e7d2c0->9rectangle] id=3: rectangle: width=0, height=0
   15: Create: [0x4e7d400->6object] id=4:
   16: Create: [0x4e7d400->9rectangle] id=4: rectangle: width=5, height=0
   17: Create: [0x4e7d4c0->6object] id=5:
   18: Create: [0x4e7d4c0->9rectangle] id=5: rectangle: width=8, height=0
   19:
   20: Object: [0x4e7d090->6circle] id=1: circle: diameter=0
   21: Object: [0x4e7d1a0->6circle] id=2: circle: diameter=10
   22: Object: [0x4e7d2c0->9rectangle] id=3: rectangle: width=0, height=0
   23: Object: [0x4e7d400->9rectangle] id=4: rectangle: width=5, height=0
   24: Object: [0x4e7d4c0->9rectangle] id=5: rectangle: width=8, height=0
   25:
   26: return 0
   27: Delete: [0x4e7d090->6circle] id=1: circle: diameter=0
   28: Delete: [0x4e7d090->6object] id=1:
   29: Delete: [0x4e7d1a0->6circle] id=2: circle: diameter=10
   30: Delete: [0x4e7d1a0->6object] id=2:
   31: Delete: [0x4e7d2c0->9rectangle] id=3: rectangle: width=0, height=0
   32: Delete: [0x4e7d2c0->6object] id=3:
   33: Delete: [0x4e7d400->9rectangle] id=4: rectangle: width=5, height=0
   34: Delete: [0x4e7d400->6object] id=4:
   35: Delete: [0x4e7d4c0->9rectangle] id=5: rectangle: width=8, height=0
   36: Delete: [0x4e7d4c0->6object] id=5:
   37: ==6348==
   38: ==6348== HEAP SUMMARY:
   39: ==6348==     in use at exit: 0 bytes in 0 blocks
   40: ==6348==   total heap usage: 15 allocs, 15 frees, 513 bytes allocated
   41: ==6348==
   42: ==6348== All heap blocks were freed -- no leaks are possible
   43: ==6348==
   44: ==6348== For counts of detected and suppressed errors, rerun with: -v
   45: ==6348== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 6 from 6)
```