

```

1 // Demonstrate references, const references, const member functions,
2 // ECE3090
3 // George F. Riley, Georgia Tech, Spring 2012
4
5 #include <stdio.h>
6 #include <iostream>
7
8 using namespace std;
9
10 // Define class A with a default constructor, non-default constructor,
11 // and a "Copy Constructor".
12 class A {
13 public:
14     A();           // Default constructor
15     A(int);       // Non-Default Constructor
16     A(const A&);  // A copy constructor is used by the compile whenever
17                 // a "copy" of an object is needed.
18     ~A();        // And a destructor
19 public:
20     void operator=(const A& rhs); // Assignment operator
21     void Print();               // Member function to print the x variable
22     void PrintC() const;       // Another print, the "const" keyword is a promise
23                                 // that the PrintC member function will not change
24                                 // any members in the object.
25     int x;                     // Single data member
26 public:
27     // Demonstrate a function returning a reference
28     int& X() { return x; }
29 };
30
31 A::A()
32     : x(0)
33 {
34     cout << "Hello from A::A() Default constructor" << endl;
35 }
36
37 A::A(int i)
38     : x(i)
39 {
40     cout << "Hello from A::A(int) constructor" << endl;
41 }
42
43 A::A(const A& a)
44     : x(a.x)
45 {
46     cout << "Hello from A::A(const A&) constructor" << endl;
47 }
48
49 A::~~A()
50 {
51     cout << "Hello from A::A destructor" << endl;
52 }
53
54 void A::operator=(const A& rhs)
55 {
56     x = rhs.x;

```

Program references.cc

```

57     cout << "Hello from A::operator=" << endl;
58 }
59
60 void A::Print()
61 {
62     cout << "A::Print(), x " << x << endl;
63 }
64
65 void A::PrintC() const
66 {
67     cout << "A::PrintC(), x " << x << endl;
68 }
69
70 // Define some subroutines to call with variations on A arguments.
71 void PassAByValue(A a)
72 {
73     cout << "PassAByValue, a.x " << a.x << endl;
74     a.x++; // Increment the x member of the passed object
75     a.Print();
76     a.PrintC();
77 }
78
79 void PassAByReference(A& a)
80 {
81     cout << "PassAByReference, a.x " << a.x << endl;
82     a.x++; // Increment the x member of the passed object
83     a.Print();
84     a.PrintC();
85 }
86
87 void PassAByConstReference(const A& a)
88 {
89     cout << "PassAByReference, a.x " << a.x << endl;
90     // a.x++; // Increment the x member of the passed object..won't compile
91     a.PrintC(); // Call the "const" print function...works perfectly
92     //a.Print(); // Call to "non-const" print function fails!
93     // Compiler error from above line. Why?
94     // references.cc: In function 'void PassAByConstReference(const A&)':
95     // references.cc:90: error: passing 'const A' as 'this' argument of
96     // 'void A::Print()' discards qualifiers
97 }
98
99 void PassAByPointer(A* a)
100 {
101     cout << "PassAByPointer, a->x " << a->x << endl;
102     a->x++;
103     a->Print();
104     a->PrintC();
105 }
106
107
108 int main()
109 {
110     cout << "Creating a0"; getchar();
111     A a0; // Default constructor called
112

```

Program references.cc (continued)

```

113     cout << "Creating a1"; getchar();
114     A a1(1); // Constructor with integer argument
115
116     cout << "Creating a2"; getchar();
117     A a2(a0); // Copy constructor
118
119     cout << "Creating a3"; getchar();
120     A a3 = a0; // Which constructor?
121
122     cout << "Assigning a3 = a1"; getchar();
123     a3 = a1; // Which constructor, if any?
124
125     // Call some of the "A" subroutines
126     cout << "PassAByValue(a1)"; getchar();
127     PassAByValue(a1); // Which constructor, if any?
128     cout << "After PassAByValue(a1)" << endl;
129     a1.Print();
130
131     cout << "PassAByReference(a1)"; getchar();
132     PassAByReference(a1); // Which constructor, if any?
133     cout << "After PassAByReference(a1)" << endl;
134     a1.Print();
135
136     cout << "PassAByConst(a1)"; getchar();
137     PassAByConstReference(a1); // Which constructor, if any?
138     cout << "After PassAByConstReference(a1)" << endl;
139     a1.Print();
140
141     cout << "PassAByPointer(&a1)"; getchar();
142     PassAByPointer(&a1); // Which constructor, if any?
143     cout << "After PassAByPointer(a1)" << endl;
144     a1.Print();
145
146     // Use the X() member function to get a reference to x
147     cout << "a1.X() = 10"; getchar();
148     a1.X() = 10;
149     a1.Print();
150
151     cout << "PassAByConstReference"; getchar();
152     PassAByConstReference(20);
153     // Why does the above compile? What does it do?
154     return 0;
155 }

```

Program references.cc (continued)