

Inheritance and Dynamic Binding

Consider the following two classes¹:

```
1 #include <iostream>
2 using namespace std;
3
4 class A {
5     protected:
6         int i;
7
8     public:
9         A(int i) : i(i) {}
10
11         int f() const { return 2 * i; }
12
13         int g() const { return 3 * i; }
14
15         virtual int h() const { return 4 * i; }
16
17         friend ostream& operator<<(ostream& o, const A& a) {
18             return o << "A( i: " << a.i << " )";
19         }
20 };
21
22 class B : public A {
23     int j;
24
25     public:
26         B(int i, int j) : A(i), j(j) {}
27
28         int g() const { return i + j; }
29
30         virtual int h() const { return 40 * i; }
31
32         friend ostream& operator<<(ostream& o, const B& b) {
33             return o << "B( i: " << b.i << ", j: " << b.j << " )";
34         }
35 };
```

¹The sole aim of this exercise is to test the understanding of inheritance and dynamic binding. To this end the names of classes and their members were chosen to be brief and not to convey any meaning. This is not an example of good programming style.

What is the output of the following program?

```
1  int main() {
2      A a1(3), a2(5);
3      cout << " a1 is " << a1 << "; a2 is " << a2 << endl;
4
5      a1 = a2;
6      cout << "after a1 = a2:" << endl
7          << " a1 is " << a1 << endl
8          << " a1.f() == " << a1.f() << endl
9          << " a1.g() == " << a1.g() << endl
10         << " a1.h() == " << a1.h() << endl;
11
12     B b(7, 2);
13     cout << " b is " << b << endl
14         << " b.f() == " << b.f() << endl
15         << " b.g() == " << b.g() << endl
16         << " b.h() == " << b.h() << endl;
17
18     a1 = b;
19     cout << "after a1 = b:" << endl
20         << " a1 is " << a1 << endl
21         << " a1.f() == " << a1.f() << endl
22         << " a1.g() == " << a1.g() << endl
23         << " a1.h() == " << a1.h() << endl;
24
25     A* pa = new A(5);
26     B* pb = new B(7, 2);
27     cout << " *pa is " << *pa << endl
28         << " pa->f() == " << pa->f() << endl
29         << " pa->g() == " << pa->g() << endl
30         << " pa->h() == " << pa->h() << endl;
31     cout << " *pb is " << *pb << endl
32         << " pb->f() == " << pb->f() << endl
33         << " pb->g() == " << pb->g() << endl
34         << " pb->h() == " << pb->h() << endl;
35
36     pa = pb;
37     cout << "after pa = pb:" << endl
38         << " *pa is " << *pa << endl
39         << " *pb is " << *pb << endl
40         << " pa->f() == " << pa->f() << endl
41         << " pa->g() == " << pa->g() << endl
42         << " pa->h() == " << pa->h() << endl;}
```