Stream Substitutability

Writing data to a file is almost as easy as printing it on the screen. Once an `ofstream` object is set up, you can use the `<<` operator with the file stream in the same way you can with the `cout` object:

```
int x = 42;
ofstream out("myfile.dat");
cout << "x->" << x << endl;   // of course this works
out << "x->" << x << endl;    // this works as well
```

Well, the question is, **why does that work?** To understand this, think back to the `write` function that you created for the `Point` structure:

```
ostream& write(ostream& out, const Point& p)
{
    out << "(" << p.x << ", " << p.y << ")";
    return out;
};
```

This works with `cout` and `cerr`, both of which are `ostream` objects.

```
Point p = {4, 2};
write(cout, p) << endl;
write(cerr, p) << endl;
```

So, what do you have to do to adapt the function so that it works with `ofstream` objects and maybe even `ostringstream` objects? The answer, perhaps surprisingly, is that **you do not have to do anything**; it already works with `ofstream` objects just as it does with `ostream` objects like `cout`, **because every `ofstream` object IS-A `ostream` object** through the **principle of substitutability**.