Assigned Reading on Object-Oriented Programming (for the Final Exam)

C++ virtual member functions are like Java methods (and unlike ordinary C++ member functions) in that they *can be overridden in derived classes to allow* ***runtime*** *polymorphism*. **<u>Read about virtual functions on pp. 269 – 71 of Sethi</u>**; these pages are included in the course reader. (It may well be a good idea to read more. For example, you might read from the beginning of Sec. 7.4 on p. 267 through the bottom of p. 271, though—assuming you have successfully completed CSCI 211 [Object-Oriented Programming in C++]—you should *already* know much if not all of that material.)

On the Final Exam there will be one or more questions, worth a total of 3 – 5 points, that test your understanding of this topic. For this/these question(s), you will be given some code and may be asked questions to test your understanding of how / whether the presence or absence of the `virtual` modifier in one or more of the function declarations affects the code's meaning or validity. For example, you might be asked about how / whether the code's meaning or validity would be affected if the `virtual` modifier were added to or removed from certain lines.

If, for instance, you were given the code shown at the bottom of p. 269 and the upper half of p. 270, then you might possibly be asked the question Sethi asks in the first sentence below that code—i.e., "What are the values of the expressions `d.testF()` and `d.testG()`?" [As Sethi explains on p. 270, the answers are `'D'` and `'B'`, respectively.]

As another example, if you were given the code shown in Fig. 7.10 on p. 271, then you might possibly be asked:

If `s` and `p` are variables of type `Shape *`, which of the following is a correct statement about the code "`s = new Circle; s->draw(p);`"?

(a) It is illegal. (b) It calls `s->Shape::draw(p)`. (c) It calls `s->Ellipse::draw(p)`.
(d) It calls `s->Text::draw(p)`. (e) It calls `s->Circle::draw(p)`.

[As Sethi explains on p. 271, (e) is the answer. But if we removed the `virtual` modifier from the declaration of `Shape::draw()` in Fig. 7.10, then (b) would be the answer.]