

Name: <i>(as it would appear on official course roster)</i>	
Umail address: _____ @umail.ucsb.edu	section 9:30am or 11am
Optional: name you wish to be called if different from name above.	
Optional: name of "homework buddy" (leaving this blank signifies "I worked alone")	

1

h07

CS56 M18

h07: HFDP: Chapter 1

ready?	assigned	due	points
true	Fri 08/31 12:30PM	Tue 09/04 09:30AM	100

You may collaborate on this homework with AT MOST one person, an optional "homework buddy".

MAY ONLY BE TURNED IN IN THE LECTURE/LAB LISTED ABOVE AS THE DUE DATE, OR IF APPLICABLE, SUBMITTED ON GRADESCOPE. There is NO MAKEUP for missed assignments; in place of that, we drop the three lowest scores (if you have zeros, those are the three lowest scores.)

Reading Assignment: HFDP is "Head First Design Patterns", the second required textbook for the course. Please Read [HFDP Chapter 1](#) along with the online reading notes. Then, answer the questions below.

During W16, the CMPSC 56 students came up with questions based on reading assignments in HFDP. Some of the questions below are based on suggestions—those questions include a "thanks").


- (10 pts) This assignment should be submitted by *scanning the pages in the correct order* to a PDF file and uploading to <https://gradescope.com>.

For more information, visit <http://ucsb-cs56-pconrad.github.io> and look for [Gradescope: Student Self Submission](#) under "topics".

Even though it is a Gradescope submission, nevertheless, *please fill in the information at the top of this homework sheet*, including your name and uemail address. Put the time your discussion section starts (9:30am or 11am) in the space indicated (the one you are registered for—even if you usually attend a different one.)

If the other two items apply, please fill them in as well. Please do this every single time you submit homework for this class.

- (Thanks to Giovanni R.). Refer to the code under the heading "*Setting Behavior Dynamically*" on HFDP p. 20 ([on-campus](#) [off-campus](#)).

Note: To see page numbers in online version, toggle HTML View button ()

- (10 pts) What parts of the code on this page allow the user to set duck behavior dynamically at run-time?
 - (10 pts) Why is this not possible if we had a concrete implementation that lives inside of the duck class?
- The author describes one of the benefits of design patterns as being that of a "shared vocabulary".
 - (5 pts) What does the author mean by "shared vocabulary"?
 - (5 pts) How is a "shared vocabulary" of design patterns useful to software developers?
 - (5 pts) Chapter 1 of HFDP focuses on refactoring some code in a Duck simulator to encapsulate various algorithms for flying and quacking, each into its own class. What is the name given to this design pattern?

2

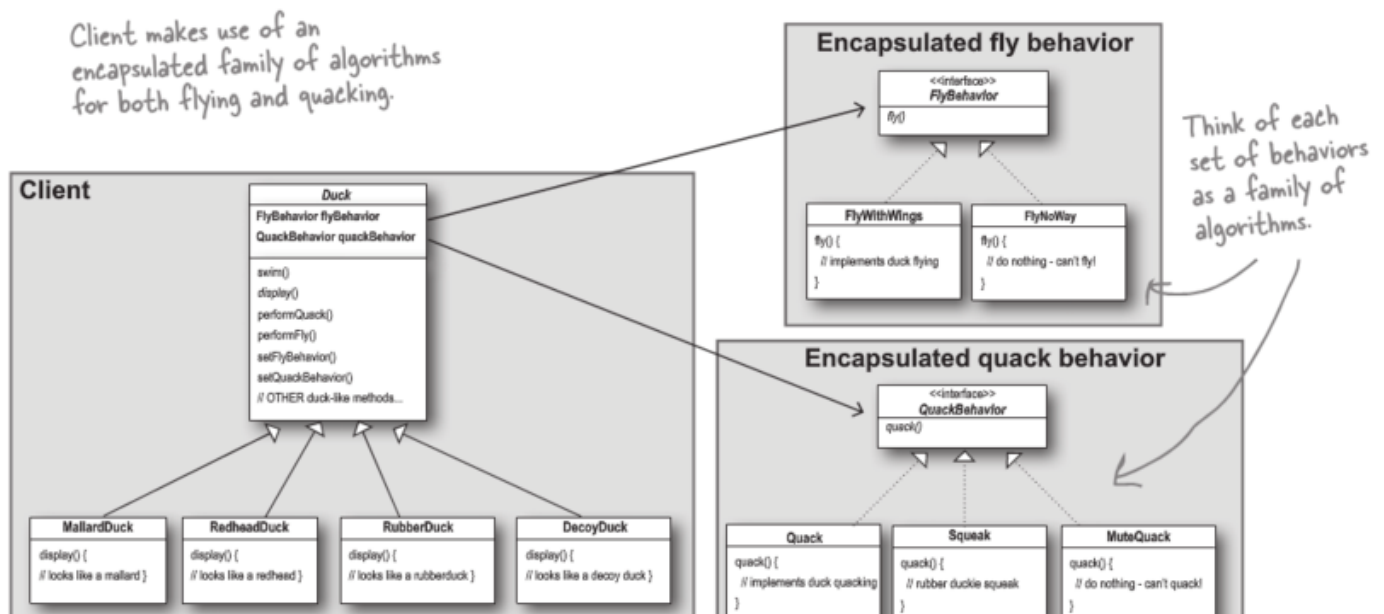
h07

CS56 M18

5. (20 pts) At a recent meetup (in 2015) sponsored by several Santa Barbara area tech employers (Procore, AppFolio, Invoca and others), OOP expert Sandi Metz gave a talk in which she reference the “Single Responsibility Principle (SRP)”, a principle expressed by OOP expert Robert C. Martin thus: “A class should have only one reason to change.”. You read more about SRP on [Wikipedia](https://en.wikipedia.org/wiki/Single_responsibility_principle).

Does the refactoring of the Duck code in Chapter 1 move the code closer to the Single Responsibility Principle? Give an answer “yes” or “no”, and then justify your answer with sound arguments that show your understanding of the code, the material in HFDP Chapter 1, and the SRP (as explained above, and on the Wikipedia page.)

6. (15 pts) On HFDP, p. 23, there is the following diagram. The authors invite you to write IS-A or HAS-A on each arrow. Please do so on this copy of the diagram. There are 11 arrows that should be labelled (only the ones inside the boxes; not the ones from the “hand written notes”).
(Note to grader: -2 for each mislabelled arrow, with a minimum grade of zero.)



7. (10 pts) (Thanks to Thien H.) The textbook authors make the point that despite design patterns being a good idea, you are unlikely to find a library or framework where you can just `import` a design pattern, the way, for example, you can `import` a data structure such as `HashMap`, `ArrayList`, or `XmlParser`. According to the authors, why is this the case?

8. The authors make a point (one that is echoed often by folks in industry) that when choosing between inheritance and composition, one of those is preferred over the other. (Note that this is a general principle—the point is not that the “less preferred” technique is worthless—just that it should be used less often, and only where it is appropriate.)

a. (5 pts) According to this principle, between “inheritance” and “composition” which is preferred?

b. (5 pts) Does the “preferred” technique correspond to HAS-A or IS-A?