Name:		
(as it would appear on official course roster)		
Umail address:	@umail.ucsb.edu	section 5pm, 6pm, 7pm
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

h04: HFJ 7,8: Inheritance, Polymorphism, Abstract Classes, JN7 Ch2 pp.33-50.

h04

CS56 F19

ready?	assigned	due	points
true	Mon 10/07 05:00PM	Wed 10/09 05:00PM	100

Printable PDF *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:

- Read HFJ Chapter 7 and HFJ Chapter 8, along with the online reading notes that go with those chapters.

Also skim JN7 Ch2 pp. 33-50. While you should read the HFJ chapters carefully, in this case, pp. 33-50 can mostly be skimmed; they are primarily for reference and I want you to know where to turn in the book if you need that material. You should, however, make sure you understand the difference between associativy and precedence, and read carefully enough that you are able find the answers to the questions given.

Then, do the problems below. Refer back to previous chapters as needed.

1. (10 pts) Please fill in the information at the top of this homework sheet, including your name and umail address. Put the time your discussion section starts (5pm, 6pm, 7pm) 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.

2. (10 pts) Based on your reading in HFJ Chapter 7, complete the following exercise from p. 179, putting a check next to the relationships that make sense.

   Note to grader: subtract 2 for each incorrect answer, with a minimum grade of zero.

✓	__ *extends* __
	Oven extends Kitchen
	Guitar extends Instrument
	Person extends Employee
	Ferrari extends Engine
	FriedEgg extends Food

✓	__ *extends* __
	Beagle extends Pet
	Container extends Jar
	Metal extends Titanium
	GratefulDead extends Band
	Beverage extends Martini

Note: *The Grateful Dead* is a band from the 1960s, and a beagle is a type of dog.

3. (10 pts) Based on your reading in HFJ Chapter 7:

   What does it mean to have a “polymorphic argument” or a “polymorphic return type” for a method? Explain with an example—but NOT using the example of Vets and Animals used in the book. Substitute your own example. Give a detailed enough description of the class hierarchy you have in mind to make it clear that you get the concept.

4. (10 pts) Based on your reading in HFJ Chapter 8:

   Briefly describe the difference between an abstract class and an interface.

5. (10 pts) What is one advantage of using an `ArrayList` over a plain old Java array (e.g. `ArrayList<Integer> nums;` vs. `int [] nums;`)?

Java code examples

Several questions on this page ask you to illustrate your answers with Java code excerpts.

Your examples do not need to be a "complete" Java classes. They can be partial Java classes with with some ellipsis (...) to show parts you are leaving out. It needs to have just enough detail to illustrate the point you are making. For example the following Java code might be used to illustrate various things (what those are is left for you to figure out.)

```
public class Bar { ... }
public class Baz { ... }
public class Foo extends Bar {
  private Baz thingy;
  ...
}
```

2

h04

CS56 F19

6. Two terms that are very important, and commonly confused are *overloading* vs. *overriding*. There is a passage in HFJ Chapter 7 that helps to explain the difference.

   a. (10 pts) Explain what overloading is, with a *specific* example of Java code.

   b. (10 pts) Explaing what overriding is, with a *specific* example of Java code.

7. There is a passage in HFJ Chapter 7 that helps to explain the difference between IS-A and HAS-A:

   a. (5 pts) Which one corresponds to *composition*?

   b. (5 pts) Which one corresponds to *inheritance*?

8. From JN7 Ch3, pp. 33-50:

   a. (10 pts) Explain the difference between operator associativity and operator precedence

   b. (5 pts) How many levels of operator precedence are there in Java?

   c. (5 pts) In Java (also in C and C++) there is an operator with three operands (a "ternary" operator). The first operand must always be an expression of a specific type. What is this type?