1. Provide definitions for each of the following terms. (Remember to cite sources, even if the source is the textbook. Wikipedia is not usable as a primary source.)

- abstraction
- encapsulation
- polymorphism
- cohesion
- coupling
- identity

Besides the definition, discuss how the term applies to the object-oriented notion of a class. Provide examples of both good and bad uses of these terms in the design of a class or a set of classes (can be code or a text/graphic example)

2. A company has asked us to design a payroll system that will pay employees for the work they perform each month. Using a level of abstraction, similar to that shown in slide 14 of lecture 4, develop a design for this system using the functional decomposition approach. You can assume the existence of a database that contains all of the information you need on your employees. For your answer, first describe the functional decomposition approach, discuss what assumptions you are making concerning this problem, and then present your design.

3. Now develop a design for the payroll system using the object-oriented approach, keeping in mind the points discussed on slides 30-32 of lecture 4. Identify the classes you would include in your design and their responsibilities. (As before, you can assume the existence of a database and that you'll be able to create objects based on the information stored in that database.) Then, identify what objects you would instantiate and in what order and how they would work together to fulfill the responsibilities associated with the payroll system.

4. Write a simple OO program in Java 8 or later that implements a Zoo full of Animals. The Animals in your Zoo are represented on slide 16 of Lecture 5. Create a class structure similar to slide 17 with at least three levels of inheritance (e.g. Animal, Feline, Cat). You may decide how many of each animal live in your zoo, but there should be at least two instances of each subclass (like Cat). Individual animals should have individual names for their instances that start with the same letter of their subclass (e.g. Charlie and Chloe for Cat).

You will also need a class for a Zookeeper. The Zookeeper will have the following responsibilities – wake the animals, roll call the animals, feed the animals, exercise the animals, shut down the zoo. Each animal will have a method that responds to this (wakeup, make noise, eat, roam, sleep). You can decide when it's appropriate to use a method at a superclass or subclass level for each animal's behavior, but there should be some behaviors that are placed at each of the three inheritance levels. In at least one case (probably for the Cats) use a random number generation to select from alternative responses to animal actions (e.g. when you ask a cat to sleep, it may randomly sleep, hiss, or run around).

When the program runs, the Zookeeper should execute each of his responsibilities in order (display a string for each action he executes), and the animals should respond by displaying strings with their name, their type, and their response to the Zookeeper. It is likely you will have a main program that instantiates the objects before they begin to act. Your program should demonstrate polymorphism by

asking your collection of animals to perform their tasks by referring to a collection of all animals at the Zoo when executing methods. Capture the text output of the final program in an out file called "dayatthezoo.out". Your submission must include your Java code, a README (described below), and this out file.

Homework/Project 1 is worth 50 points – 10 per question 1-3, 20 for the program in number 4. There is no extra credit element for this assignment.

Grading Rubric:

Questions 1 through 3 will be graded on the quality (not quantity) of the answer. Remember for question 1 we need definitions and examples. A solid answer will get 10 points, missing elements or poor quality answers will cost -2 points each, missing answers completely will be -4 points.

Question 4: 20 points possible. Documentation = 5 points (-1 or -2 for incomplete or missing elements). Execution demonstrated in out file = 5 points (-1 or -2 for missing expected functionality or output). Code quality = 10 points (-1, -2, -3 for issues including poor commenting, procedural style code, or logic errors; -10 if not an object-oriented solution as requested).

Submissions:

Questions 1-3 must be submitted via a PDF file and must include all names of team members.

Question 4 should be submitted via a GitHub Repository URL, including the code, the outfile, and a README Markdown file with the title of the project, the names of team members, any comments on or assumptions made for the development, and any special instructions to run the code. Your Java code should be well structured and commented, and citations and/or URLs should be present for any code taken from external sources. Comments should focus on what's being done in the code, not on how, unless the method in question is unclear or obscure. You may not directly use code developed by other student teams, although you may discuss approaches to the problems, and may wish to credit other teams (or class staff) for ideas or direction.

Homework/Project 1 is Due at 12 Noon on Friday 1/31.

Assignments will be accepted late for one week. There is no late penalty within 4 hours of the due date/time. In the next 24 hours, the penalty for a late submission is 5%. After that the late penalty increases to 15% of the grade. After the one week point, assignments will not be accepted. The team may submit the assignment using a late pass from each of the team members. Only one late pass can be submitted per project, and it will extend the due date/time 24 hours.

Use e-mail or Piazza to reach me or the class staff regarding homework questions.