Half-Exam 2 Name:

No note-sheets, calculators, etc. on this exam. Please read the whole exam before you get started.

1. (30 points) Consider the following problem description: In a family pet office, each veterinarian handles two kinds of pets: dogs and cats. Each dog or cat has a name, a height and a weight. Dogs have a breed. For each pet there is a single owner, but one owner can have multiple pets. Owners have names and phone numbers. At any one time, certain pets have appointments, so some pets have no appointment, but others have a single appointment. An appointment is for a specified date and time. Each veterinarian has a name and an identification number, and one veterinarian works each appointment.
    a. ***Circle*** the unique nouns and ***box*** the verbs. Ensure you find them all! It is critical for the next parts.

    b. ***List*** all nouns that should not appear as classes in a domain-level diagram.

    c. ***Use*** the information from the paragraph to draw a domain-level class diagram for this system. Draw generalizations and associations with multiplicities. Include simple attributes and operations (but without any types) if they are given in the statement.

2. (20 points) ***Write a short code example*** illustrating control coupling

3. (40 points) In a competition application, students are on teams. Each student enters their name, phone number, home address, etc. There are different sorts of students. Some students are high-school students and other students are college students, but students can change from high-school to college during runtime! When validating the information a student enters, high-school students must enter a phone number and college students must enter an address. Students who are high-school students should be able to change to college students at run-time. At a minimum, ***include*** classes for the teams and students. ***Draw*** a UML class diagram capturing how the strategy pattern would be used in this case. Be sure to ***mark*** abstract classes and interfaces as well as to ***show*** associations (with arrows), inheritance relationships, multiplicities, methods, and attributes.

4. (10 points) ***Name*** the class in your design that is programmed to an interface and ***write a very brief code snippet*** illustrating how it programs to the interface.