This assignment consists of implementing the classes and the class relationships represented in a UML class diagram, instantiating objects from the classes, pushing data into the objects with constructors, and pulling the data back out using operators. The lab extends Lab #8 by adding aggregation and composition and uses pointers and dynamic memory (chap 10). The vet example illustrates the syntax needed to complete the assignment.

**Assignment**

Modify the code from Lab 8 by adding a composed Date class and an aggregated Address class as illustrated in the UML diagram on the next page. Complete the program with `driver.cpp`.

1. **General class features:**
   a. Make all member variables (i.e., attributes) private.
   b. Make all member functions and friend functions public.
   c. Initialize all member variables via constructor calls.
   d. Define each class in a separate .h file (remember the #ifndef and #endif preprocessor statements)

2. Add Address and Date classes (the classes are similar to those used in the vet example and you may copy them from the example but notice that the relations with Employee are reversed from those in the vet example).

3. The Employee class
   a. add a setAddress function as specified in the UML diagram on the next page (see the first setShots function in dog.h of the vet example)
   b. add variables to implement the class relationships with Date and Address (note that these variables are not shown in the diagram but are assumed from the connector lines). Take care to set the variable implementing the aggregation relation with Address to NULL (see the Dog constructor in the vet example)
   c. update the constructor to reflect the additional data that is passed to the Date constructor
   d. update operator<< to print the birth date and the address
   e. define a destructor that deletes the aggregated Address object
   f. make sure you understand the use of “new” in the driver

4. The SalesEmployee class
   a. update the constructor to reflect the additional data that is passed to the Salaried constructor

5. The SalariedEmployee class
   a. update the constructor to reflect the additional data that is passed to the Employee constructor

6. The WagedEmployee class
   a. update the constructor to reflect the additional data that is passed to the Employee constructor

7. Complete the program with the included drivier.cpp
Program Submission

- Upload the six header files (employee.h, salariedemployee.h, wagedemployee.h, salesemployee.h, date.h, and address.h) to BlackBoard (make a new directory and upload the files to that directory)
- Make sure that the files are correctly named
- Remember to press the “Submit” button
- Please do not zip the files