Lab 4

Due April 18th

(100 pts)

You may work with a partner if you want. Turn in one version with 2 names on it. Do not forget your partner's name. Or work alone. Your choice.

Problem 1 (10 pts):

Create 2 classes. The first class (heretofore referred to as Class A, which, of course, you will change to be more descriptive when you write the class) must have as one of its fields something of the type second class (heretofore referred to as Class B, which, of course, you will change to be more descriptive again when you write the class). In other words, give a good and clear example of composition. Do not use any of the examples we used in class. Make sure that:

a. The reason you are using class composition makes sense (e.g., a card class and a deck class that is composed of a list of card objects makes complete and utter sense). I want some example that makes sense (that we have not used in our class or in the powerpoints).
b. Both classes have at least 2 fields
    a. In Class A, one of the fields must either be of the type of ClassB, or it must be an array of Class B (again, think the deck field in the DeckofCards class).
c. Both classes have at least 2 constructors.
    a. In Class A, one of the constructors must create an object of the type of Class B and set the appropriate field to that object.
d. The fields should all be private, so you will need appropriate getter and setter methods for your classes.
e. One method in Class A will do something with and return something that was calculated or concatenated with a field in the field in the field that is of type Class B (got that? I'm going for using Class B's field in the Class A method).
f. Write a toString method for both classes
g. Test all this.
    a. Make objects of both classes
    b. Test the setters and the getters
    c. Test the toString method APPROPRIATELY!
    d. Test your other method(s).
h. Comment the heck out of this so we know why you think this is a good example of why you'd use composition.

Problem 2 (10 pts):

Create 2 classes. The first class (heretofore referred to as Class A, which, of course, you will change to be more descriptive when you write the class) will be a super class and the second class (heretofore referred to as Class B, which, of course, you will change to be more descriptive again when you write the class) will be its subclass. In other words, give a good and clear example of inheritance. Do not use any of the examples we used in class. Make sure that:

a. The reason you are using class inheritance makes sense (e.g., a circle class and a cylinder derived class makes perfect sense). I want some example that makes sense (that we have not used in our class or in the powerpoints).
b. Both classes have at least 2 fields
    a. Because we haven't learned protected yet, the fields and methods in your super class must be public so the subclass can access them.
c. Both classes have at least 2 constructors.
    a. In Class B, one of the constructors call the first constructor in Class A, and the other constructor must call the second constructor in Class A.
d. Class A will have at least 2 methods
    a. one that is unique and generalized for all things of type Class A and for all things derived from Class A
    b. one that is a general method (which in step e you will be ***overriding***)
e. Class B will have a method that ***overrides*** the method written for Class A above.
f. Write a toString method for both classes
g. Test all this.
    a. Make objects of both classes
        i. Make at least 2 objects of type Class B, using both constructors.
    b. Test the toString method APPROPRIATELY for all objects!
    c. Test your other method(s) (especially the one that was overridden)
h. Comment the heck out of this so we know why you think this is a good example of why you'd use inheritance.

Problem 3: Using this to call constructors: (5 pts)

Remember the watch class you created for problem 1 in lab 3? Modify it so that within the constructors you use *this* to call the constructors. Make sure you test each constructor by creating 4 different watch objects and then printing them out.

Problem 4 (25 pts): Composition (tictactoe):

Create a class for a Player. Include private fields for the player's name, the player's piece as a char ('x' or 'o'), and a Boolean value indicating whether the Player is a human or a computer.

Create 2 constructors for this player – one that automatically sets the player's name to be "computer" and the Boolean value to be false, and another that takes 2 parameters as input – the name in the form of a string and the Boolean value indicating whether it is a computer or not.

Note that the constructors don't set the piece to either 'x' or 'o' – this will be set randomly later in the actual game class.

Create 5 methods – the 3 getters (getName(), getisPerson(), and getPiece()), one setter (setPiece()), and the toString method.

Now create a class for the tictactoe game. This class should have 3 fields (all private): player1, player2, and a 3x3 board fo characters.

Create 2 constructors – one that creates a computer player and a human player (I named mine 'Annie' but you can make a default name of whatever you like). Use 'this' to call the 2nd constructor. The other constructor should take a String for the name of the player and the constructor should create a computer player and a human player with the name of the input parameter. The constructor should also call a method makeBoard that sets the original board to 3x3 arrays of '_' characters.

So write the method makeBoard(as described above).

Write a private method setPlayerPieces() that returns nothing and randomly sets player1's piece to 'x' or 'o' and player2's piece to the other piece.

Write a method (private) called compPlays that takes the character (x or o) as an input parameter and randomly places that character in an empty square on the board. This is mimicking the computer placing a piece on the board in a very unintelligent manner.

> Extra Credit (1-8 pts) – this is the dumbest computer playing function ever. ANYTHING would be better. Make the computer smarter at playing for at most 8 extra credit points.

Write a method (private) called personPlays that also takes the character to be placed on the board. This method asks the user to input the row in which they wish to place the piece, then asks for the column. If that square is empty, it places the piece. Otherwise it keeps asking the user to input the row and then the column until it is able to place the piece on the board.

Write a private method checkWin that takes an input character c and returns a Boolean value indicating whether that character c has won or not. It checks the board to see of that character occurs completely across a row, a column, or either diagonal. If it does, it should return true. Otherwise it should return false. (Note: DO NOT MAKE A HUGE IF CONDITION!!! – you must use nested loops to receive full credit here.)

Now write the method playGame that returns a Boolean value true if someone has won, and false if no one has won and the board is full (i.e., it is a tie). This method should first call setPlayerPieces. It should randomly choose who plays first (the computer or the person). It should then call the compPlays and the personPlays methods, checking after each for whether the current player won or not. If someone won, it should print out the player who won (using the toString method of the Player) and return true. Regardless, it should print out the current board. It should continue to play until either someone wins (returning true) or the board is full (returning false).

Note: you may want to write a method to check whether the board is full or not, returning a boolean value.

And write a toString method for the tictactoegame, including both players and the board.

Finally, create a tictactoe object, and then for that object call the playGame() method.

****

Problem 5: Inheritance: Insects and inheritance: (30 pts)

We're going to create a hierarchy of Bugs.

In order to do that you will need to start by creating a class for Insects. This is your superclass. Each insect should have:

- an armor field, indicating the amount of armor it has (an int),
- and fields for the x and y coordinates of where the insect is located.
- Clearly your insect class needs a constructor that sets the armor amount and the original x and y coordinates.
- It will also need a method that reduces the armor by an amount (an int). If the armor is less than 0, the x and y coordinates should both be set to a negative number (-1).
- It will need a toString method that prints out the type (Insect), the amount of armor the insect has, and the insect's x and y coordinates.
- Finally it will need a method that returns a Boolean value. This function will make more sense when we make other subclasses. But this method should be named isAnt() and right now it should return false. That's the whole function.

Next you want to create a class for Ants. This class is a subclass of Insects. Thus its constructor should set the armor and the x and y coordinates of the Insect. This class will also have fields (whose values you should set with passed in parameters) for:

- the amount of damage it can inflict (an int),
- the amount of food it uses (an int),
- and a name field (a String).

- In addition, this class should also have a method isAnt() that returns a Boolean value, and, in this case, the Boolean value should be true.

Now you want to create a class for a Bee. This class is a subclass of Insects. Thus its constructor should set the armor and the x and y coordinates of the Insect. This class will also have fields for:

- The name ("Bee") (a String)
- (Note that you'll need to call the Insect constructor with the appropriate values, and these values should be passed in as parameters to the bee's constructor).
- There should also be a sting method that takes as an input parameter something of type Ant. The method should reduce the armor of the ant by 1 (using the reduceArmor method from the Insect class).
- There should be a method that moves the bee to a new location (within one square of where it is currently, using its x and y coordinates. So, for instance, it can move 1 square directly up, one square down, one square to the left, one square to the right, or one square in any of the 4 diagonal directions. This method should generate random numbers to move the bee, and then reset the x and y coordinates.
- And it should have a toString method that returns the word "bee" plus the amount of armor the bee has.

Now you want to create some specialized ants. You'll want to create a class for Harvester Ant, which is a subclass of the Ant class. The Harvester Ant should call the Ant's constructor, setting the amount of armor, the x and y coordinates, and the name of the insect ("Harvester").

The harvester Ant needs a toString method that prints out the word “Harvester” and the amount of armor the ant has.

And finally, you’ll want to create a ThrowerAnt, which is a subclass of Ant. The ThrowerAnt needs to set the armor, the x and y coordinates using parameters passed into the ThrowerAnt’s constructor, along with the name (using “Thrower”). In the constructor, the damage field should be set to 1.

- The ThrowerAnt needs a method thowsAt, which takes as an input parameter a Bee. It should reduce the bee’s armor by whatever the ThrowerAnt’s damage amount is. (You’re using the reduceArmor method in the Insect class.
- And, of course, the ThrowerAnt needs a toString method that returns a string with “Thrower” and the amount of armor the thrower ant has.

Now write a main method. In it create at least one Bee, one Harvester And, and one Thrower Ant. Test the different methods in all the classes. Make sure you test the isAnt method with both the bee and the ants. Make sure you have the bee sting each ant and then print out each Ant. Use the moveTo method to move the bee and print it out. Make sure you have the throwerAnt attack the bee using the throwsAt method, and then print out the bee.

****

Problem 6 (20 pts): Keypad entry:

Remember in the old days, when phones didn’t have full keyboards, and you had to type in using your phone’s keypad. To get an ‘a’, you’d use the 2 to get a ‘d’, you’d use the 3, etc. Your phone did a surprisingly good job of “disambiguating”, or determining which word you actually wanted based on the numbers you typed in. Believe it or not, this technology was originally developed for people with muscle control difficulties (you knew I’d bring this back to people with disabilities, didn’t you?). They had trouble with fine motor control, so they used 8 large buttons and were able to enter information into the computer that way. The technology was then sold to Sprint (I think – I get my phone companies mixed up) and the rest is history. As far as I know, the technology is still being used in Europe (they built some intelligence into the system so that when a number sequence can result in more than one word, the system uses surrounding words and the user’s history of word preference to make a better choice about which word to pick).

To be clear, on your phone keypad, the alphabets are mapped to digits as follows: `ABC(2)`, `DEF(3)`, `GHI(4)`, `JKL(5)`, `MNO(6)`, `PQRS(7)`, `TUV(8)`, `WXYZ(9)`.

Your Turn: You’ll need to create 2 classes, and a main class.

The first class will be the wordpair class. It will have as fields a word string and a string that represents that word’s corresponding number sequence.

Its constructor will take a string holding a word, and set both the field for the word, and the field for the numerical representation of the word. I wrote a method that translated the word into its corresponding numeric sequence (as a string) and returned that string.

The second class, a wordlist class will have as a field, an array of objects created from the wordpair class.

Its constructor will take as an input parameter the name of a text file containing a list of words. The constructor will read in the list of words and, for each word, it will create a wordpair object with both the word and the word's corresponding numeric string. It will add that wordpair object to the array field. (Note two things: you'll have to read the entire file to find out the length of the array you will need to create, then close the file, and create a new array of the desired length, and then read the file again and create the wordpair objects and add them to the new array. It sets the array field to this new array. Also, at the top of this class, you'll need to include the following:

```java
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;
```

You will also need to two methods: one that takes as input a String representing a word, and returns a string representing its numerical representation. If the word isn't found in your array field, it should return "Word not found". The other method should take as input a String representing a numerical combination and it should return the corresponding word. Mine printed out all possible matches and then returned the last matching word.

Finally, this class should have a method that should return nothing, but should go through the array field and print out all words with matching numerical strings.

You will then need to write a main class in which you create a wordlist object, and test all the methods in the class.