Homework 1

For problems 1 through 4, explain why the code as shown is almost certainly not what the programmer intended, and how it should be fixed to work the way the programmer probably had in mind.

1. (10 pts) What is wrong with the following program and how should it be fixed?

```
public class MyClassA {
  int v = 12;

  public MyClassA (int pV) {
    v = pV;
  }

  public static void main (String args []) {
    MyClassA m = new MyClassA ();
  } // end main
} // end class MyClassA
```

The programmer most likely intended to retain the ability to create an instance of MyClassA without arguments, as indicated by the creation of variable m. However, the program will fail to compile, because it is expecting arguments for the MyClassA initialization call. When initializing an object of MyClassA, a parameter of type int is required. While it is true that the Java compiler can create a “default” no-argument constructor, it will only do so in the absence of explicitly defined constructors. As such, the programmer needs to create another constructor, this time without parameters.

2. (10 pts) What is wrong with the following program and how should it be fixed?

```
public class MyClassB {
  int v = 12;

  public void MyClassB (int pV) {
    v = pV;
  }

  public static void main (String args []) {
    MyClassB m = new MyClassB (23);
  } // end main
} // end class MyClassB
```

The programmer most likely intended to create a constructor that accepts an int argument, but created a method instead. As such, the program fails to compile, because it was not expecting arguments for the MyClassB initialization call. Constructors do not have a return type. Since there is no explicitly defined constructor in this class, the Java compiler created a default, no-argument constructor. In order to remedy this, the programmer can simply remove the return type, “void,” from the “constructor.”

3. (10 pts) What is wrong with the following program and how should it be fixed?

```
1   public class MyClassD {
2     public static void main (String args []) {
3       MyClassC m = new MyClassC (23);
4     } // end main
5   } // end class MyClassD
6
7   class MyClassC {
8     int v = 12;
9
10    public MyClassC (int pV) {
11      int v = pV;
12    }
13
14  } // end class MyClassC
```

The programmer most likely intended to store the value of pV in the class variable v, but stored it in a local variable of the same name instead. Still, the program will compile without error. The value passed during initialization of the MyClassC object is lost at the end of the constructor method. This is because a variable, v, is being declared within the constructor method, hiding the class variable of the same name. If the programmer meant to assign the value of pV to the class variable, the programmer needs to remove the type assignment for the variable inside the constructor (i.e. line 11 should read “v = pV;”).

4. (10 pts) What is wrong with the following program and how should it be fixed?

```
1   public class MyClassE {
2     public static void main (String args []) {
3       MyClassF m = new MyClassF (23);
4     } // end main
5   } // end class MyClassE
6
7   class MyClassF {
8     int v = 12;
9
10    private MyClassF (int pV) {
11      v = pV;
12    }
13
14  } // end class MyClassF
```

The programmer most likely intended to have the ability to instantiate MyClassF from other classes. The program will fail to compile because the constructor for MyClassF is not visible to MyClassE. While the use of private constructors is valid for singleton classes, such classes cannot be instantiated outside themselves, ensuring only one object is ever created. To remedy the error, the constructor scope of MyClassF needs to be either removed or set to public.

5. (10 pts) Given all the problems identified in problems 1 through 4, explain in detail why the following code works, ie, compiles without errors or warnings.

```
public class MyClassG {
  public static void main (String args []) {
    MyClassH m = new MyClassH (23, true);
  } // end main
} // end class MyClassG

class MyClassH {
  int v = 12;

  public MyClassH (int x, boolean b) {
    this (x);
  }

  private MyClassH (int pV) {
    v = pV;
  }

} // end class MyClassH
```

The program compiles without errors or warnings, because the appropriate scopes and arguments have been met. When the program runs, it will start with attempting to instantiate an object of MyClassH using the arguments of 23, an int, and true, a boolean. Since MyClassH has a constructor that accepts these two parameters and is visible to MyClassG, the program can make the call to the appropriate constructor. The MyClassH(int,bool) constructor has a call to another constructor that accepts only an integer parameter, to which it passes the int value it received from the call in MyClassG. Even though the MyClassH(int) constructor is private, since the call is being made within the class, it is visible. That said, the constructor called by MyClassG is able to call the other constructor of MyClassH.

6. (10 pts) Explain why the following class hierarchy is not reasonable:

- DefenseDepartment
    - General
        - Private

The class hierarchy is not reasonable because it does not reflect an “is-a” relationship. A soldier with the rank of Private could eventually become a General, but a Private is not, nor ever will be, a General. Likewise, a General is not a DefenseDepartment.

7. (10 pts) Give at least one example of a reasonable field for each of the following classes in the following class hierarchy. Be sure that the field is at the right level in the hierarchy.

- Vehicle
    - Car
    - Airplane
        - Passenger
        - Fighter
        - Bomber
    - SpaceShip

- Vehicle
  - amountOfFuel:double
    - Car
      - drivetrain:Drivetrain
    - Airplane
      - numberOfEngines:int
        - Passenger
          - numberOfPassengers:int
        - Fighter
          - caliberOfCannons:int
        - Bomber
          - numberOfBombs:int
    - SpaceShip
      - inOrbit:bool

8. (10 pts) Give at least one example of a reasonable method for each of the following classes in the following class hierarchy. Be sure that the method is at the right level in the hierarchy. Constructors, getters and setters don't count for this problem.

- Vehicle
    - Car
    - Airplane
        - Passenger
        - Fighter
        - Bomber
    - SpaceShip

- Vehicle
  - refuel()
    - Car
      - lowerWindow(Window):void
    - Airplane
      - takeOff():bool
        - Passenger
          - boardPassengers():void

        - Fighter
          - fireCannons(Duration):void
        - Bomber
          - dropBombs(int):void
    - SpaceShip
      - returnToEarth():void

9. (10 pts) Are a Private and a Platoon in an encapsulation or an inheritance relationship? Explain

A Private and a Platoon are in an encapsulation not an inheritance relationship. A Private is not a Platoon, nor is a Platoon a Private, so they cannot be in an inheritance, or “is-a,” relationship. A Platoon can have many Privates, so they would be in a “has-some” relationship. All the Privates belonging to the Platoon are “encapsulated” in the Platoon.

Another reason for encapsulation can be explained with an example. When a commander assigns a platoon a task, the commander does not need to know what each private is doing, just that the platoon is doing the task. This is similar to starting a car: the operator does not need to know the details of how the car is starting just that turning the key (or pushing the button) starts the car.

10. (10 pts) Present reasonable parent and child classes for the class Tree (the biological kind). Give a short explanation for why the classes you are proposing are in good parent-child relationships.

A reasonable parent for the class Tree could be the class Plant, because all trees are plants. A reasonable parent for the class Tree could be the class Coniferous or the class Deciduous. Both of these are types of trees and should have all the same traits and behaviors as trees, though the implementation may be different. For example, deciduous trees shed leaves in the winter while coniferous trees keep them.

Grading Rubric:

Attribute	Meets	Does not meet
Problem 1	**10 points** Explains why the code as shown is almost certainly not what the programmer intended. Explains how it should be fixed to work the way the programmer probably had in mind.	**0 points** Does not explain why the code as shown is almost certainly not what the programmer intended. Does not explain how it should be fixed to work the way the programmer probably had in mind.
Problem 2	**10 points** Explains why the code as shown is almost certainly not	**0 points** Does not explain why the code as shown is almost certainly not

	what the programmer intended. Explains how it should be fixed to work the way the programmer probably had in mind.	what the programmer intended. Does not explain how it should be fixed to work the way the programmer probably had in mind.
Problem 3	**10 points** Explains why the code as shown is almost certainly not what the programmer intended. Explains how it should be fixed to work the way the programmer probably had in mind.	**0 points** Does not explain why the code as shown is almost certainly not what the programmer intended. Does not explain how it should be fixed to work the way the programmer probably had in mind.
Problem 4	**10 points** Explains why the code as shown is almost certainly not what the programmer intended. Explains how it should be fixed to work the way the programmer probably had in mind.	**0 points** Does not explain why the code as shown is almost certainly not what the programmer intended. Does not explain how it should be fixed to work the way the programmer probably had in mind.
Problem 5	**10 points** Given all the problems identified in problems 1 through 4, explains in detail why the code works, ie, compiles without errors or warnings.	**0 points** Given all the problems identified in problems 1 through 4, does not explain in detail why the code works, ie, compiles without errors or warnings.
Problem 6	**10 points** Explains why the class hierarchy is not reasonable.	**0 points** Does not explain why the class hierarchy is not reasonable.
Problem 7	**10 points** Gives at least one example of a reasonable field for each of the classes. The field is at the right level in the hierarchy.	**0 points** Does not give at least one example of a reasonable field for each of the classes. The field is not at the right level in the hierarchy.
Problem 8	**10 points** Gives at least one example of a reasonable method for each	**0 points** Does not give at least one example of a reasonable method

	of the classes. The method is at the right level in the hierarchy. Does not include constructors, getters and setters.	for each of the classes. The method is not at the right level in the hierarchy. Includes constructors, getters and setters.
Problem 9	**10 points** Explains inheritance and encapsulation correctly and in sufficient detail given the example provided.	**0 points** Does not explain inheritance and encapsulation correctly and in sufficient detail given the example provided.
Problem 10	**10 points** Presents reasonable parent and child classes for the class Tree. Gives a short explanation for why the classes you are proposing are in good parent-child relationships.	**0 points** Does not present reasonable parent and child classes for the class Tree. Does not give a short explanation for why the classes you are proposing are in good parent-child relationships.