Question 1. (3 points) Give an algorithm for making hot tea.

Question 2. (4 points) One of the simple examples in the book is the class `Counter`. The code is for reference only to help you fill in the blanks in the paragraph below.

```
public class Counter {

   private int tally;

   public int count() {
      return tally;
   }

   public void stepCount() {
      tally = tally + 1;
   }

   public Counter() {
      tally = 0;
   }
}
```

Choose the words or phrases from the following list **only** that *best* complete the sentences below so that they are true.

> **assignment, argument, boolean, class, class definition, client, conditional, constructor, declaration, double, expression, integer, instance variable, instance, local variable, loop, message, method, name thingy, parameter, precedence, return statement, return value, scratch space, state, String, this, type, void**

In the definition of class Counter, the line of code reading "`private int tally;`" is the declaration of a/an ___________________________ of the class. When a new Counter object is created, the ___________________________ is executed and `tally` is initialized. The ____________________ statement "`tally = tally + 1;`" increases the value of `tally`. In that statement, "`tally + 1;`" is an example of a/an _____________________ .

Question 3. (6 points) Suppose you have been hired to design a software system to support a video rental store. Describe two classes that you would use to build this system. For each class, list at least 3 properties and 3 responsibilities that you would include in the class. For each property, list the type of the property and for each responsibility, state if it is a query or command. No Java programming is required to answer this question

Class #1 Name ______________________________

Property	**Type**

Responsibility	**Query or Command?**

Class #2 Name ______________________________

Property	**Type**

Responsibility	**Query or Command?**

Question 4. (3 points) Describe the differences between the specification and implementation of a class.

Question 5. (5 points) Trace the execution of the following code by recording the state and output produced when the code executes. Please show all changes to the state (values of variables) by crossing out values as they are updated.

```
int num = 5;
int denom = 5;
double result = 0.0;
while (num > 0 ) {
    result = (double) num / denom;
    System.out.println(result);
    num = num - 1;
}
```

Trace:

State **Output**_____________

num =

denom =

result =

Question 6. (15 points) Suppose you are asked to develop a payroll system for a company. You are given the following partially written class definition for Employee objects to start with.

```
/** A class representing an employee of the company */
public class Employee {
    private double hours;        // hours worked in a week
    private double rate;         // hourly pay rate
    private String name;         // employee name
    private Employee manager;    // employee’s manager

    /** Get the number of hours worked thus far in the week */
    public double getHours () {
        return hours;
    }

    /** Set this employee’s manager */
    public void setManager (Employee employeeManager) {
        manager = employeeManager;
    }

    /** Construct an Employee object with the given employee name
      *  and hourly pay rate. */
    public Employee (String employeeName, double hourlyRate) {
        name = employeeName;
        rate = hourlyRate;
        hours = 0.0;
    }
 }
```

(a) (5 points) Write the specification and implementation of a method named `updateHours` to increase an employee object’s hours worked. The method should take as a parameter the hours worked and update the hours for the employee. As a safeguard, if the hours worked is less than 0.0, the method should return the Boolean value false and not update the hours for the employee. If the hours worked is greater than or equal to 0.0, the method should update the hours and return the Boolean value true.

Question 6. (cont.) (b) (5 points) Write the specification and implementation of a method named `getWeeklyPay` to calculate and return the weekly pay for an employee object. An employee is paid at the hourly rate for the first 40 hours worked in the week and paid at 1.5 times of their hourly rate for additional hours worked over 40 hours.

(c) (5 points) Suppose we enter the following statements in DrJava's interactions window to create and use a pair of Employee objects. Draw object diagrams showing the state of each object and the relationship(s) among the objects after these statements have been executed.

```
Employee boss = new Employee (“B. Bedrock”, 45.25);
Employee foreman = new Employee (“F. Flintstone”, 19.50);
foreman.setManager(boss);
boss.updateHours(14.3);
```