1. Analyze the class Cake presented and write all concepts and ideas you can remember regarding what is a class and how we can define and use it. Make a comment line by line.

public class Cake {  
 public static final boolean *IS\_GOOD* = true; *//constant class variable* private String name; *//instance variable* private double cost; *//instance variable  
 //sets a value for the instance variable name* public void setName(String str) {  
 name = str;  
 }  
 *//return the current value of instance variable name* public String getName() {  
 return name;  
 }  
 *//sets a value for the instance variable cost  
 //we use this. to make reference to the instance variable  
 //when we have a parameter with the same name of the instance variable* public void setCost(double cost) {  
 this.cost = cost;  
 }  
 *//return the current value of the instance variable cost* public double getCost() {  
 return this.cost;  
 }  
 *//constructor - calls the Cake constructor with 2 parameters* public Cake(){  
 this("", 0);  
 }  
 *//constructor method - initialize the instance variables* public Cake(String name, double cost) {  
 setName(name);  
 setCost(cost);  
 }  
}

1. Identify: class variables (scope and type), instance variables (scope and type). What is the purpose of the class variable in this example? How can we access the class variable from another class? How can we access the instance variable from another class? Write a StoreApp class to have a main method and in that method creates an ArrayList of Store, add three Store objects into the ArrayList and print the current value of nextId.
2. public class Store {  
    public static int *nextId* = 101;*//public class variable - automatically generates ids* private String name;*//private instance variable* private String type;*//private instance variable* private int id; *//private instance variable  
    //constructor* public Store(String storeName, String storeType) {  
    name = storeName;  
    type = storeType;  
    id = *nextId*; *//id received the current nextId number* ++*nextId*;*//update the nextId class variable - to next id* }  
     
    public int getId(){  
    return id;  
    }  
   }

import java.util.ArrayList;  
  
public class AppStore {  
 public static void main(String args[]){  
 ArrayList<Store> list = new ArrayList<Store>();  
 Store s = new Store("Store 1", "Food");  
 list.add(s);  
 s = new Store("Store 2", "Beauty");  
 list.add(s);  
 s = new Store("Store 3", "Culinary");  
 list.add(s);  
 for(Store elem: list){  
 System.*out*.println(elem.getId());  
 }  
 System.*out*.println("Next id available:" + Store.*nextId*);  
 }  
}

1. Rewrite the class Pet to have its constructors properly overloaded.

public class Pet {  
 private String name;  
 private int age;  
  
 public Pet() {  
 this("Unnamed", -1);  
 }  
  
 public Pet(String petName, int yearsOld) {  
 name = petName;  
 age = yearsOld;  
 }  
  
 public String toString() {  
 return name + ", " + age;  
 }  
}

1. Explain each line of the main method. For the method writeHTMLFile, explain how many parameters it receives and what the method does.

import java.io.File;  
import java.io.IOException;  
import java.io.PrintWriter;  
  
public class HTMLFileWriteSample {  
 *//method received a PrintWriter object and a string to be added into an HTML basic format template* public static void writeHTMLFile(PrintWriter printer, String innerHTML) {  
 *//write all the strings in the PrintWriter object passed as a parameter* printer.println("<!DOCTYPE html>");  
 printer.println("<html>");  
 printer.println(" <body>");  
 printer.println(" <p>" + innerHTML + "</p>");  
 printer.println(" </body>");  
 printer.println("</html>");  
 }  
 public static void main(String[] args){  
 try{  
 String fileName = "simple.html";  
 *//creates a PrintWriter object name filePrinter for the simple.html file* PrintWriter filePrinter = new PrintWriter(new File(fileName));  
 *//calls the method that write a basic HTML in the filePrinter  
 writeHTMLFile*(filePrinter, "Hello <b>HTML</b> world!");  
 *//closes the filePrinter* filePrinter.close();  
 *//creates a PrintWriter object name systemOutPrinter that prints to the terminal stream (System.out)* PrintWriter systemOutPrinter = new PrintWriter(System.*out*);  
 *//calls the method that prints the basic html to the terminal, since the file is the terminal  
 writeHTMLFile*(systemOutPrinter, "Hello <b>HTML</b> world!");  
 *//close the file related to the terminal stream* systemOutPrinter.close();  
 }catch(IOException exp){  
 exp.printStackTrace();  
 }  
 }  
}

1. What is the difference between adding “import java.util.Scanner;” and “import java.util.\*;” in a program?

First one imports only Scanner, second one imports all the classes that are inside of .util package.