Week 1 – Class variables and methods

Explain how class (static) variables and methods differ from their instance counterparts. Give an example of a meaningful class that contains at least one class variable and at least one class method, beside the instance variables and methods. Explain why using a class variable and method rather than an instance variable and method would be the correct choice in the example you select.

Week 2 – The super keyword

What does the super keyword represents and where can it be used? Give an example of a superclass and subclass. Be sure to make all the instances variables of the super class private. Include at least one constructor in each class and ensure that the constructor of the subclass calls the constructor of the superclass. Also include a toString method in both classes that returns the values of the instance variables with appropriate labels. Ensure that the toString method of subclass calls the toString method of the superclass so that the string returned contains the values of all the inherited instance variables.

Week 3 – Exception propagation

What is exception propagation? Give an example of a class that contains at least two methods, in which one method calls another. Ensure that the subordinate method will call a predefined Java method that can throw a checked exception. The subordinate method should not catch the exception. Explain how exception propagation will occur in your example.

Week 4 - GUIs and event handlers

Give an example of a program that creates a Swing based GUI with at least one button, one group of radio-buttons and several textfields. Some of the textfields should be for input and others for output. Make the output textfields uneditable. When the button is clicked, the input fields should be read, some calculation performed considering the active radio-button and the result displayed in the output textfield(s).

For example, the simple GUI may have an input textfield for circle radius, an output textfiled for result, a button Calculate and a group of two radio buttons: Area and Perimeter. When Calculate button is clicked, the radius value is read and the area or perimeter is calculated and displayed in the result based on the active radio-button.

Week 5 - Good practice of GUI design

GUI design practice states that any GUI should include elements to allow/perform five basic tasks: (1) user input, (2) system output, (3) user guidance, (4) interaction control and (5) organize GUI appearance. In this context:  
  
a. Think of a simple GUI (such as Hotel Registration GUI, Conference Payment GUI, etc.). Your GUI is: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
b. Choose the appropriate components and other resources that should be used to achieve each of the first four basic elements (1-4) of the considered GUI. Using paper and pencil design the GUI layout, scan it and attach it to your post. For each component indicate whether it is used for user input, system output, user guidance or interaction control.  
c. Describe how it works and give details about user – GUI interaction.  
d. Identify the layout manager(s) which is (are) appropriate to use for your GUI elements organization.

Week 6 – Writing recursive methods

Consider a simple recursive method (Java code written by yourself or taken from Web) such as calculating the sum of the first n natural numbers, calculating the number of digits of an integer number, calculating 2 to the power of n, etc. Try choosing one different from that of any posted thus far.

a. Show method code and indicate what it does and what represent its parameter(s) and the return value;

b. Identify the base case(s) and indicate how each recursive call simplifies the problem and makes progress towards the base case;

c. For the chosen method, identify the circumstances of infinite method calls.

Week 7 – Generic classes

The syntax of generic classes and methods was not added until version 5.0 of Java. Nonetheless, early versions of the language had collections classes such as an array list, or a linked list which were able to store objects, instances of different classes.

a. How were those classes implemented without the generic syntax?

b. Give an example of a program that uses the non-generic version of a class and the equivalent program that uses the generic version.

c. How do the two implementations differ?

d. Why is having the syntax of generics better? Explain and illustrate with a method of the example class.

Week 8 – JCF Map

Java Collection Framework (JCF) defines the interface Map and offers three common implementations for this interface: HashMap, TreeMap and LinkedHashMap.

a. Give an example of a real application that may use a Map implementation as its main support data structure.

b. Explain what represent the Key and the Value for the considered example.

c. Which of the three Map implementations is most appropriate for the considered example? Explain.

d. Write one or two lines of Java code to define and instantiate an object of type Map for the example of real application.