CE4703 Computer Software 3 – Programming Project

1. Overview

Your task is to write a program for circuit analysis, i.e. a program that can model circuits of resistors under DC power. The operations that your program should be able to perform are the following:

a) Resistance of the overall circuit and of the individual components of the circuit
b) Given a voltage applied to the overall circuit, the resulting voltage at the individual components.
c) Given a voltage applied to the overall circuit, the resulting current flowing through the overall circuit and through individual components of the circuit.
d) Given a list of resistors and a target resistance, work out the resistor network (definition given below) with the closest overall resistance to the target resistance. This operation should produce the solution circuit as return value.

2. Instructions

Carefully follow these steps (in the order given – it will simplify your task):

a) Develop an ADT Resistor, i.e. write down specifications for this ADT. Your ADT must support at least the following operations (feel free to add any operations you deem required/desirable):
   - Creation of resistors with a given resistance and a name/label (String). A newly created resistor will have a voltage of 0V applied (and consequently will also have a current of 0A).
   - Setting the voltage applied to the resistor (this method should also automatically calculate the resulting current through the resistor).
   - Getting the voltage currently applied to the resistor.
   - Getting the current currently flowing through the resistor.
   - Getting the name/label of the resistor
b) Implement you ADT in the Java programming language.
c) Develop an ADT Circuit, i.e. write down specification for this ADT. In this project, a circuit has at most two components. Each component is again a circuit. Your ADT should support (at least) the following operations:
   - Create circuit with given component(s) and label/name (String)
   - Get a String representation of the circuit. Use '|' for parallel and '-' for series connection.
   - Get overall resistance.
   - Get resistance of component one.
   - Get resistance of component two.
   - Set overall voltage (also calculates the overall current and set the component voltages).
   - Get overall voltage.
   - Get voltage across component one.
   - Get voltage across component two.
   - Get overall current.
   - Get current through component one.
   - Get current through component two.
   - Get the label/name of the circuit
   - Find a sub-circuit by name, i.e. the user passes in the name of a circuit and you return the object within the circuit that has that name (return type should be type of your base class).
d) Implement your ADT Circuit in the Java programming language. Use multiple classes for this ADT: An abstract base class (circuit) that defines the common interface and three derived classes:
   - One for circuits with a single component (this single component has to be a resistor),

  - One for circuits connected in series (has to have two components, where each component is of the base class type)
  - One for circuits connected in parallel (has to have two components, where each component is of the base class type)

e) Circuit Analysis: consider the following recursive construction rules for resistor networks:
  - A single resistor is a resistor network
  - A resistor network in series with a resistor is a resistor network
  - A resistor network in parallel with a resistor is a resistor network

  Find a strategy to create resistor networks as per this definition from a list of resistor and write your strategy down in English (dynamic data structures such as stack, lists or queues, are great for this purpose). To simplify construction, start with the first resistor in the list and only add on the following resistors. For example, if given the list of resistor {R1, R2, R3}, then following resistor networks are created ('-' indicates series, '|' indicates parallel):

  R1, R1-R2, R1|R2, R1-R3, (R1-R2)-R3, (R1|R2)-R3, R1|R3, (R1-R2)|R3, (R1|R2)|R3

f) Translate your strategy into pseudo-code. Do this in an iterative process as discussed in the lecture: Keep refining your pseudo-code until you are satisfied that you can translate each step of your pseudo-code into Java code. Add a method to your circuit analyser that implements this method.

g) Add a method to your circuit analyser that accepts a list (either a dynamic structure or an array) of resistors (as per you ADT in step a/b) and a target resistance. This method will find the resistor network that is closest in overall resistance to the target resistance. The return value is an object that contains the resistor network (i.e. an object of the derived classes single component circuit, series circuit or parallel circuit).

h) Add the following methods to your circuit analyser class:
  - Methods to create the circuits displayed at the end of this description (one method per circuit)
  - A user interface method (use of JOptionPane class is sufficient – there is no need to create a full GUI) that prompt the user to enter a number to select "Fixed circuit" or "Analyse circuit".
  - If "Fixed circuit" is selected, prompt the user which of the circuits should be used (1 or 2) and provide him with menu options to
    i. Get a list of component labels/names
    ii. Get a string representing any component (use '|' for parallel and '-' for series), where the component is identified by label/name
    iii. apply an overall voltage to the circuit
    iv. get resistance of any component (identified by label/name)
    v. get voltage across any component (identified by label/name)
    vi. get current through any component (identified by label/name)

- If "Analyse circuit" is selected, prompt the user for the number of resistors to be used, the value of the resistance for each resistor and the target resistance. Then invoke your method as created in steps e, f and g and display the string representation of the resistor network.

3. Marking & Deliverables

The project accounts for 30% of the overall module mark. Please form groups of 3 (three) students to undertake the project (groups can be formed across any combination of courses). Please let me know your grouping by Thursday, 4.11.2010. Any students not notifying me about their group by this deadline will be assigned into groups – no changes will be possible after this date. In general, all group members will get the same marks. If a group member does not collaborate properly within the group, please let me know: in this case, group members will receive marks according to their contribution (no contribution = 0 marks).

Please submit your solution as a single zip archive via the link provided on the module's web page. **Deadline** for submission is **Friday, 26.11.2010** – any requests for extensions must be made **before** the deadline expires.

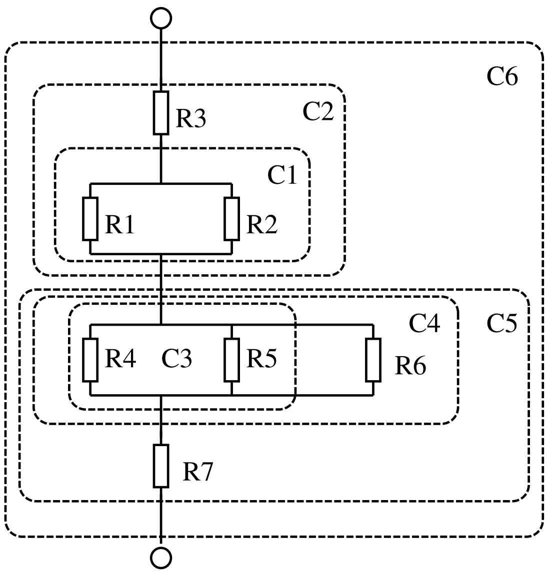
A complete solution consists of the following:
- Specifications for ADT Resistor and for ADT Circuit
- Implementation of ADT Resistor and ADT Circuit in the Java programming language using the Eclipse IDE. Your code needs to be well formatted and commented (include javadoc comments as well as comments to explain the operation of your code)
- Description of your circuit analysis method in plain English and a translation in pseudo-code (as per steps e and f above).
- Implementation of the circuit analyser class in the Java programming language using the Eclipse IDE. Your code needs to be well formatted and commented (include javadoc comments as well as comments to explain the operation of your code)

Please use a single project for all coding and submit the entire workspace (the folder you chose when Eclipse is starting up) as a zip archive. Your workspace should have the name xxxx-yyyy-zzzz, where xxxx, yyyy and zzzz are the student ID numbers of the group members.
The following marks are associated with each of the solution's components:

Component	Marks (30)
Specifications of ADT Resistor and Circuit	7
Implementations of ADT Resistor and Circuit	13
Circuit Analysis method description (English & pseudo-code)	4
Implementation of circuit analyser class	6
Total:	30

4. Miscellaneous

Circuit 1:



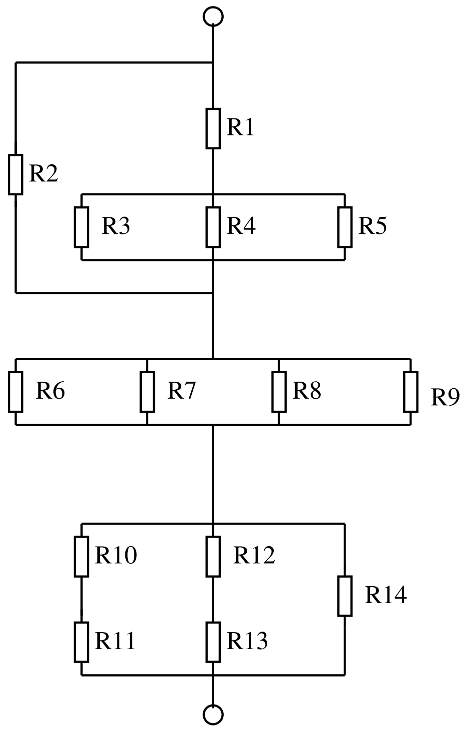
Resistors have the following values:
R1 = 50Ω, R2 = 25Ω, R3 = 30Ω, R4 = 150Ω, R5 = 75Ω, R6 = 100Ω, R7 = 40Ω

This circuit is created by first creating one object each of single component circuit for each resistor. Then
- create component C1 taking R1,R2 in parallel
- create C2 taking R3 and C1 in series,
- create C3 taking R4 and R5 in parallel
- create C4 taking C3 and R6 in parallel
- create C5 taking C4 and R7 in series
- create C6 taking C2 and C5 in series

In String form this results in:

((R1|R2)-R3)-(((R4|R5)|R6)-R7)

Circuit 2:



Resistors have the following values:
R1 = 50Ω, R2 = 25Ω, R3 = 30Ω, R4 = 150Ω, R5 = 75Ω, R6 = 100Ω, R7 = 40Ω, R8 = 60Ω, R9 = 50Ω, R10 = 20Ω, R11 = 40Ω, R12 = 30Ω, R13 = 70Ω, R14 = 80Ω

Note: Neither Circuit 1 nor Circuit 2 are resistor networks as per the above definition.

Useful formulae:

Symbols: Resistance R, Voltage U, Current I

U = R * I (from this follow R = U/I and I = U/R)

If R1 and R2 are connected in series, overall resistance R = R1+R2;
If R1 and R2 are connected in parallel, overall resistance R = (R1*R2)/(R1+R2)

Any number of resistors connected in any way with resulting resistance X, will behave exactly like a single resistor with the same resistance X, i.e. it doesn't matter if you use a single resistor with 100Ω or two resistors in series with 50Ω each.

If two resistors are connected in series, the current through them will be equal. Thus (following from U = R*I), the voltage across each resistor will be in the same ratio as their resistances (the resistor with the higher resistance will have a higher voltage).

If two resistors are connected in parallel, the voltage across each resistor will be the same. Thus (following from I = U/R), the current through each resistor will be in the inverse of the ratio of their resistances (the resistor with the higher resistance will have a lower current).