System re-engineering

Imagine that you are where you will hopefully find yourself in a few months from now: you are a graduate employee, put into the position being in charge of reengineering a legacy system in Java with 150 classes grouped into 10 clusters (10 classes in each cluster). Please assume that the code of the legacy system is copied to a repository on your own PC. You are asked to develop a detailed plan to reengineer this legacy system. This plan should contain five sections (**each section carries 20 marks**):

1. Code inspection In order to gain a broad understanding of the target legacy system, without using any tools, please:

- discuss how to use one chosen reengineering pattern to investigate the code functions for a cluster (5 marks for one pattern and 5 marks for system functions), and
- assuming the code in one cluster represents one component of the legacy system, discuss how to use one chosen reengineering pattern to investigate the legacy system's architectural features (5 marks for one pattern and 5 marks for system architecture).
- 2 - 4 pages, including at most 2 pages of diagrams.

2. Static analysis In order to analyse the legacy system statically, please

- develop and explain one algorithm to calculate the Cyclomatic Complexity for a method (5 marks for algorithm and 5 marks for explanation), and
- develop and explain one algorithm to calculate the Halstead Complexity for a class (5 marks for algorithm and 5 marks for explanation).
- 2 - 4 pages, including at most 2 pages of diagrams.

3. Dynamic analysis In order to develop a code slicer, please:

- write an algorithm with explanation to compute the Program Dependence Graph for a method (5 marks for algorithm and 5 marks for explanation); and
- write an algorithm with explanation to compute a backward slice for a given node in a method (5 marks for algorithm and 5 marks for explanation).
- 2 - 4 pages, including at most 2 pages of diagrams.

4. System visualisation In order to develop a visualiser, to visualise each cluster of classes, please:

- draw a diagram for the design of the visualiser and explain its main components (5 marks for algorithm and 5 marks for explanation to components), and
- discuss how to identify any two types of defects (must specify two defects) in code, based on visualised class structure (5 marks for identification technique and 5 marks for illustrating detection of two types of defects).
- 2 - 4 pages, including at most 2 pages of diagrams and visualisations.

5. Reengineering Code inspection, static analysis, dynamic analysis and system visualisation are basic techniques in reengineering. In order to plan an overall process for reengineering the legacy system, please

- develop a diagram of workflow using the above techniques as steps (4 marks),
- discuss how to use code inspection in the overall process (4 marks),
- discuss how to use static analysis in the overall process (4 marks),
- discuss how to use dynamic analysis in the overall process (4 marks), and
- discuss how to use system visualisation in the overall process (4 marks).
- 2 - 4 pages, including at most 2 pages of visualisations.

END