SE433/333
Individual Assignment #1

Description:

This assignment is to evaluate a project quality based on:

- Volume Metrics: Lines of Code, Number of Files/functions/Variables.
- Object Oriented: CK Metrics.

Peter Drucker, a famous management consultant, states: “If you can’t measure it, you can’t manage it”. It may not be possible for a project team to successfully manage a project if they are not able to accurately measure all the aspects of that project. Metrics have played a vital role in software industries and, as a result, failure rate of software project has decreased considerably due to the use of software project management tools and techniques. Nowadays, more efficient, robust and quantitative measures for effective software requirement, analysis, design, development, quality assurance, integration, deployment and support are in practice. The quality of software system should be measured during each phase of software system development.

In this context, CK-Metric suite is an initiative to examine the various aspect of object oriented system design such as size, coupling, cohesion and inheritance. Their metric suite to explore the bad software design consist of:

• Weighted Methods per Class (WMC)	=	CountDeclMethod
• Depth of Inheritance Tree (DIT)	=	Max Inheritance Tree
• Number of Children (NOC)	=	Count Class Derived
• Response For a Class (RFC)	=	CountDeclMethodAll
• Coupling Between Objects (CBO)	=	CountClassCoupled
• Lack of Cohesion of Methods (LCOM)	=	PercentLackOfCohesion

Understand is a commercial tool that calculates various metrics for your code during build cycles and warns you, via the Problems view, of 'range violations' for each metric. This allows you to stay continuously aware of the health of your code base. You may also export the metrics to HTML for public display or to CSV or XML format for further analysis. Note that although metrics can be useful in a software development effort, they should not take the place of good taste and experience. Also, metrics are more useful as indicators of unhealthy code than they are as indicators of healthy code, i.e. a codebase with many range violation warnings is probably an indication that the code needs to be refactored but no range violation warnings does not necessarily mean that the code is good.

You can Request for Educational Licence using your Depaul email :

License Request link https://scitools.com/student

Tasks:

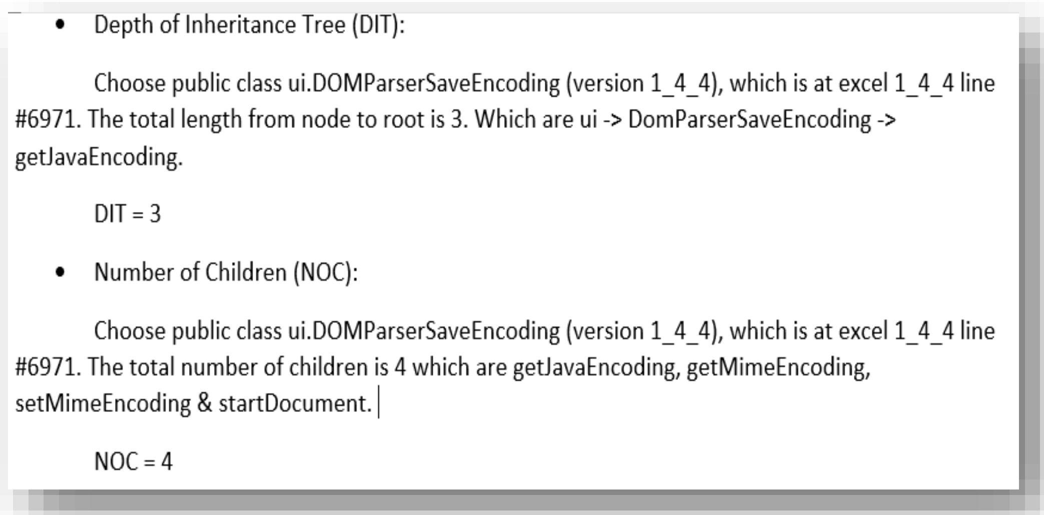
- Select one of these open source systems (4 versions from a system) **Or** search for a small/medium (=< 100 000 LOC) open source project (Java), which has at least 4 releases/versions (v1.0, v1.1, v2.0 etc.). <u>(You cannot choose *npp* system since it is given as a template example for the assignment.)</u>
    - Xerces-J (http://archive.apache.org/dist/xml/xerces-j/)
    - GanttProject (https://code.google.com/p/ganttproject/downloads/list)
    - Apache Ant (http://ant.apache.org/srcdownload.cgi)
    - Apache Tomcat (http://tomcat.apache.org/download-80.cgi)
    - jEdit (http://www.jedit.org/index.php?page=download)

- Use the tool *Understand* to calculate the metrics for each of the 4 releases/versions. The results should be saved in an excel file. Use the values of these metrics to plot the variation of metrics throughout the evolution of the software.

An example with some needed graphs for a given project was provided under Content -> Assignments -> Assignment 1.

- Use these graphs to **evaluate** the evolution of the project from quality perspective, and use them to give insights about any possible **recommendations** for developers to improve the project in the future.
- Give an example for each of these metrics *(Weighted Methods per Class (WMC), Depth of Inheritance Tree (DIT), Number of Children (NOC), Response For a Class (RFC), Coupling Between Objects (CBO), Lack of Cohesion of Methods (LCOM))* from the chosen system (any version) and show how it was calculated (if the project that you chose does not contain an example, you can provide an example from another project)

  Here is an example of the DIT and NOC metrics from a project:



- Depth of Inheritance Tree (DIT):

Choose public class ui.DOMParserSaveEncoding (version 1_4_4), which is at excel 1_4_4 line #6971. The total length from node to root is 3. Which are ui -> DomParserSaveEncoding -> getJavaEncoding.

DIT = 3

- Number of Children (NOC):

Choose public class ui.DOMParserSaveEncoding (version 1_4_4), which is at excel 1_4_4 line #6971. The total number of children is 4 which are getJavaEncoding, getMimeEncoding, setMimeEncoding & startDocument.

NOC = 4

Submission:

- One CVS or Excel file for the metrics (including metrics values for each version + average of metrics) and graphs
- One **PDF file** for the **evaluation** of the evolution of the system, the **recommendation** for developers and **examples of metrics** from the system
- If you chose another system to evaluate you also need to provide links to the source code

Submit the files in D2L (Assignment 1) **by 09/29/2022** 23:59 PM. No late submission will be accepted.