**Assignment 1 FINAL REPORT**

**Title: Analyzing the Effect of Size on Maintainability in Java Projects**

**Group-6  
 Hemanth Kumar Raju Yerramaraju**

**Sahithya Gangarapu**

**Introduction**

Software maintainability is an important aspect of software engineering that can help to determine how software systems can be fixed, upgraded, extended, or adjusted. To improve software development process by understanding the maintainability of project size is important for developers and businesses. In my report, I have examined the relation between java project size and its maintainability. For this project I have chosen java programming language due to the widespread use of various application areas like mobile apps, corporate systems, and online applications.

To confirm the systematic and structured analysis I have employed Goal-Question- Metric (GQM) framework to define clear research goals, choose appropriate metrics to give answers, and formulate the relevant questions. By utilizing this technique I can ensure that my overall conclusion will be precise, relevant and useful in the field, and well structured. In this project my primary goal is to examine the java project size and its maintainability relationship. By understanding this relationship development team can make informed decisions in the software development process and help to create more maintainable code.

To address the report goal I have formulated the following questions: Is there a significant correlation between the number of lines of code (LoC) in a Java project and its level of maintainability? By analyzing this question I have discovered the pattern which can help me to guide the best practices development to manage and sustain the large scale java project. In order to assess effectively maintainability I have identified many metrics which include Lines of Code (LoC), Coupling Between Objects(CBO), Tight Class Cohesion(TCC), and Response for a Class(RFC) to support my investigation.

CBO measures the level of interdependence between the classes to impact the class maintainability. TCC can evaluate the level of cohesion between the class methods that influence the maintainability. RFC can count the number of processes that are imposed to receive response messages by the class.

In this report I aim to explore the connection between project size (measured in LoC) and maintainability in Java projects. By utilizing the GQM framework and relevant metrics, I strive to reveal insightful patterns and trends that can inform the development of efficient approaches for managing and sustaining extensive Java projects.

**Data set description:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name** | **Repository URL** | **Description** | **Size (LoC)** | **Developers** | **Age (Years)** |
| **eclipse.jdt.ls** | [**https://github.com/eclipse/eclipse.jdt.ls**](https://github.com/eclipse/eclipse.jdt.ls) | **Java language server for IDEs** | **15,287** | **71** | **6** |
| **intellij-sdk-docs** | [**https://github.com/JetBrains/intellij-sdk-docs**](https://github.com/JetBrains/intellij-sdk-docs) | **Documentation for IntelliJ Platform SDK** | **12,893** | **186** | **8** |
| **glowroot/glowroot** | **https://github.com/glowroot/glowroot** | **Easy to use, very low overhead, Java APM** | **32537** | **20** | **9** |
| **google/guava** | **https://github.com/google/guava** | **Google core libraries for Java** | **388778** | **273** | **8** |
| **mall** | **https://github.com/macrozheng/mall** | **The project is an e-commerce system consisting of a front-end mall system and a back-end management system, implemented using SpringBoot+MyBatis and deployed with Docker containers.** | **100691** | **1** | **1** |

**Tool description**

In object oriented programming Chidamber and Kemerer Java Metrics(CKJM) tool is a specifically designed software utility to analyze the java based computing and project metrics. This tool can help to measure various software maintainability aspects like Coupling Between Objects(CBO), Tight Class Cohesion(TCC), and Response for a Class(RFC) among the other relevant metrics.

The main reason for choosing CKJM tool is to analyze the specialization of handling java projects and calculate Chidamber and Kemerer metrics. It is directly relevant for research offers and inquiry relationships between maintainability and size within the java based projects. Tool is a command line utility of a user interface which is easy to install and operate. These tools can accept the java bytecode files in class format and allow the user to produce desired metrics in a clear manner. This simplicity can help to obtain the measurements for analysis without requiring extensive knowledge of complex tools. Moreover, the CKJM tool is open-source and comes with a detailed user guide that helps users understand how the tool works and the metrics it offers. The transparency in the process of acquiring metrics ensures that the obtained measurements are reliable and trustworthy for analytical purposes. By allowing the users CKJM tool can offer to choose specific metrics for compute. In this report I have focused on CBO, TCC, and RFC metrics which are fundamental for understanding maintainability in the java based projects. This flexibility can allow the researchers to select relevant metrics to analyze, and tailoring tools for specific requirements. The CKJM reliability and dependability are supported by the widespread frequent and adoption usage in academic investigation. Its extensive acknowledgment and utilization by the research community to provide additional validation of metrics reliability. The CKJM software metric tool is accessible by subsequent Uniform Resource Locator:

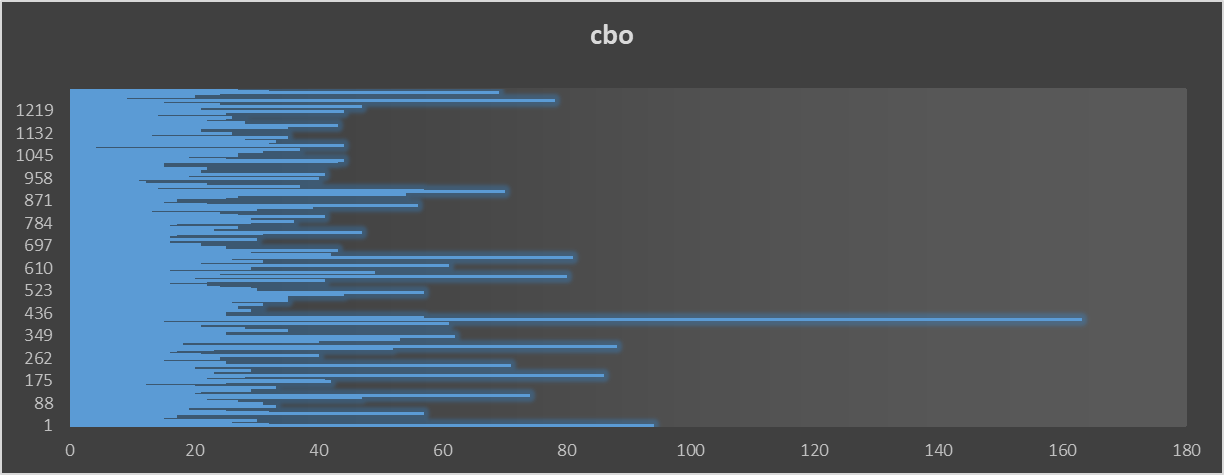
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

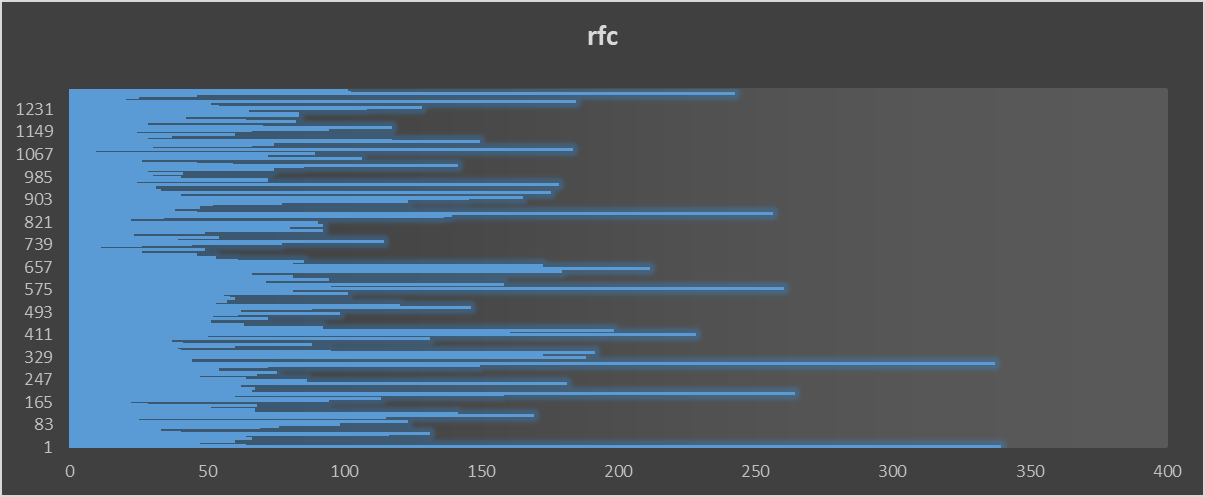
Overall,CKJM is a specialized software tool designed for analyzing Java projects and calculating Chidamber and Kemerer metrics. This tool's user-friendly interface, compatibility with Java bytecode files, transparency, metric selection flexibility, and widespread usage make it a reliable and valuable tool for studying maintainability in Java projects. Researchers can utilize CKJM to understand the connection between software metrics and maintainability, enhancing software quality.

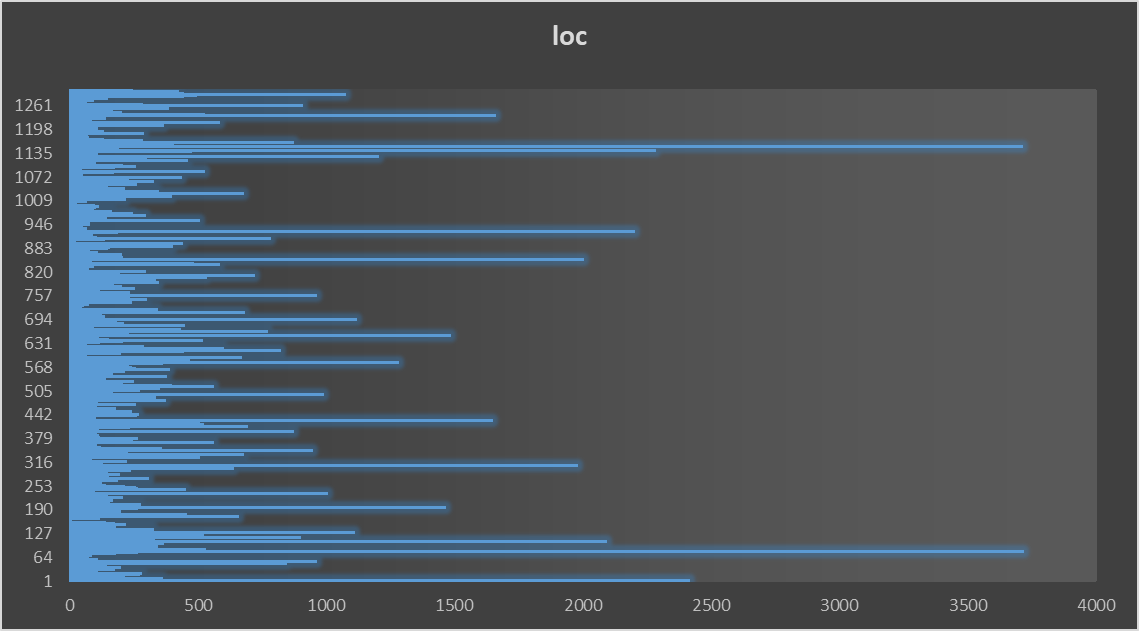
**Project Results**

**Project1 : eclipse.jdt.ls**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **CBO** | **RFC** | **LOC** |
| **Max** | **36** | **106** | **468** |
| **Mode** | **2** | **0** | **6** |
| **Median** | **3** | **4** | **16** |
| **Std Deviation** | **6.42502** | **16.24659** | **57.03431** |
| **Average** | **5.898734** | **11.04219** | **39.90506** |

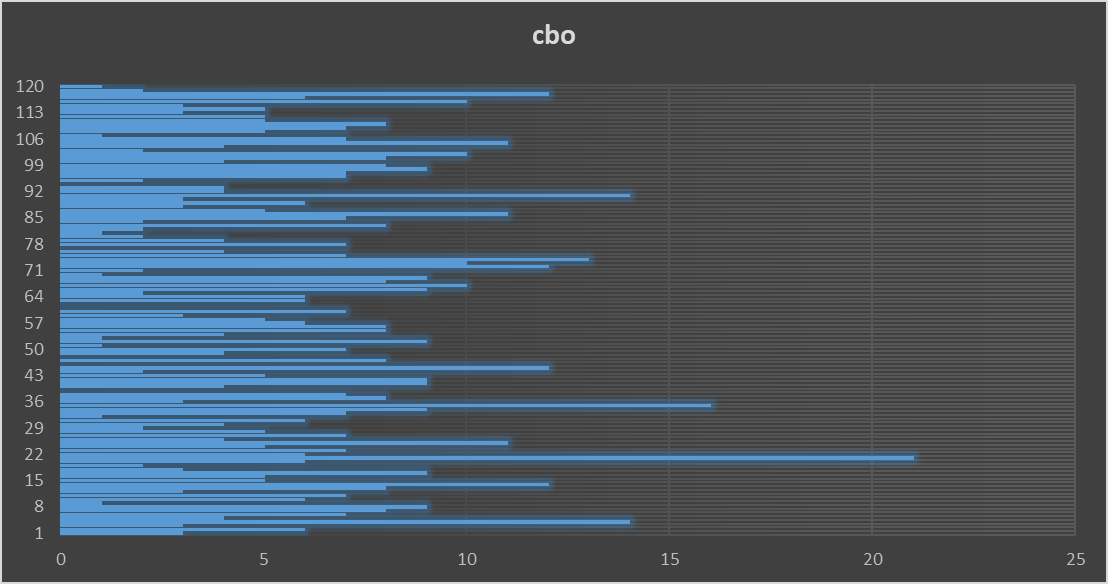
****

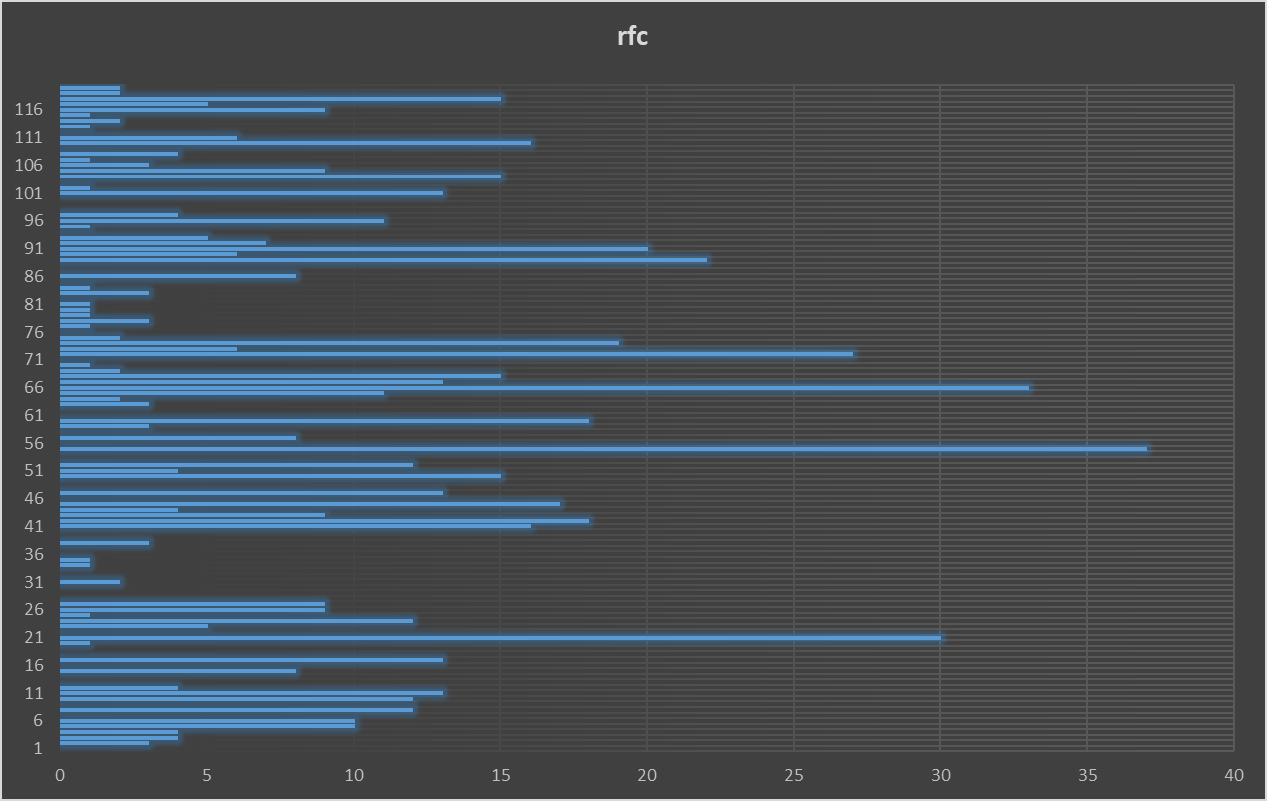
****

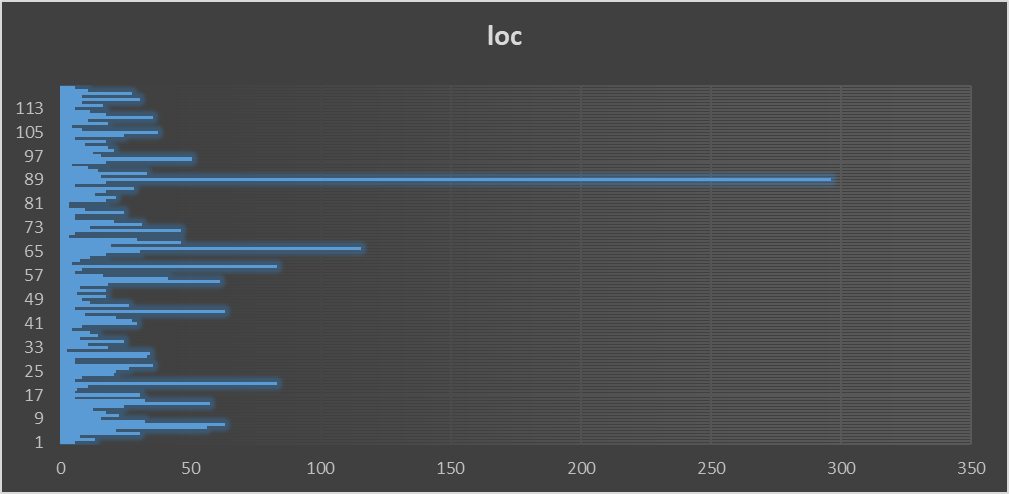
****

**Project 2: intellij-sdk-docs**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **CBO** | **RFC** | **LOC** |
| **Max** | **21** | **37** | **296** |
| **Mode** | **7** | **0** | **5** |
| **Median** | **6** | **2** | **16.5** |
| **Average** | **5.8** | **5.625** | **22.35** |
| **Std Deviation** | **3.760762** | **7.524474** | **30.98512** |

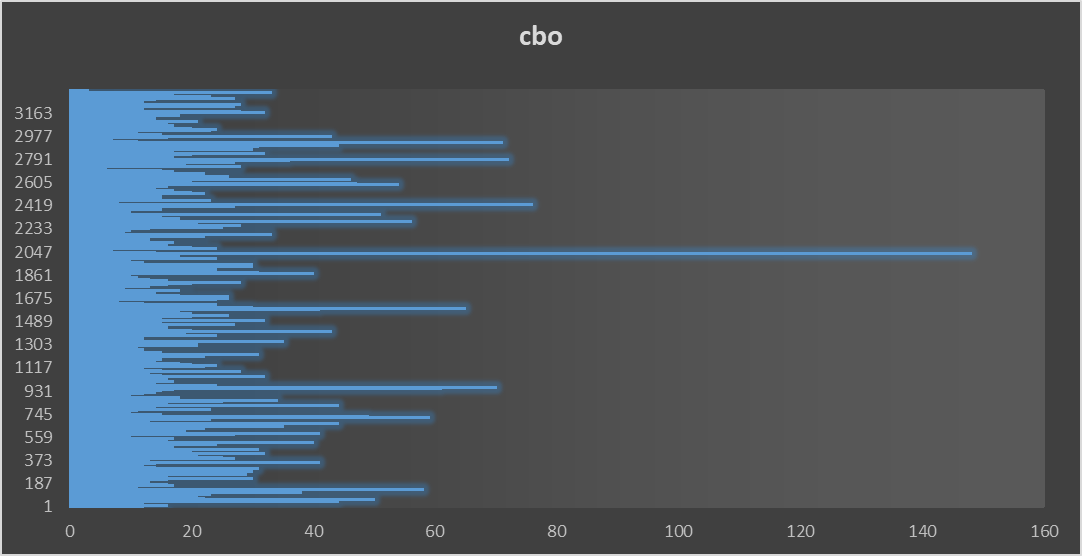
****

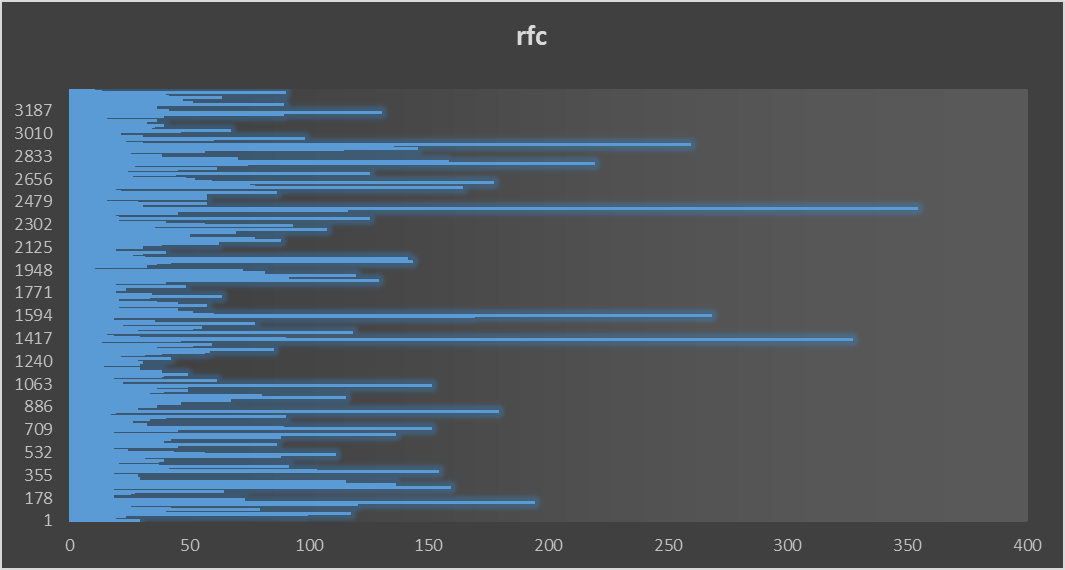
****

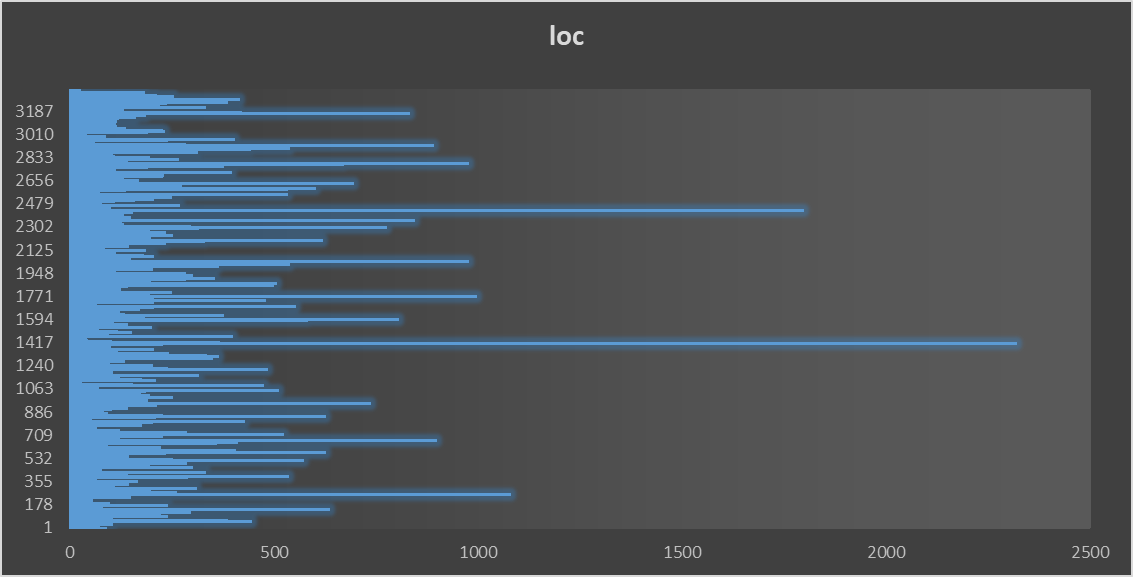
****

**Project3: glowroot**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **CBO** | **RFC** | **LOC** |
| **Max** | 148 | 354 | 2318 |
| **Mode** | 1 | 0 | 5 |
| **Median** | 3 | 3 | 12 |
| **Std Deviation** | 7.398608 | 21.13084 | 100.397 |
| **Average** | 5.341398 | 8.866487 | 40.30615 |

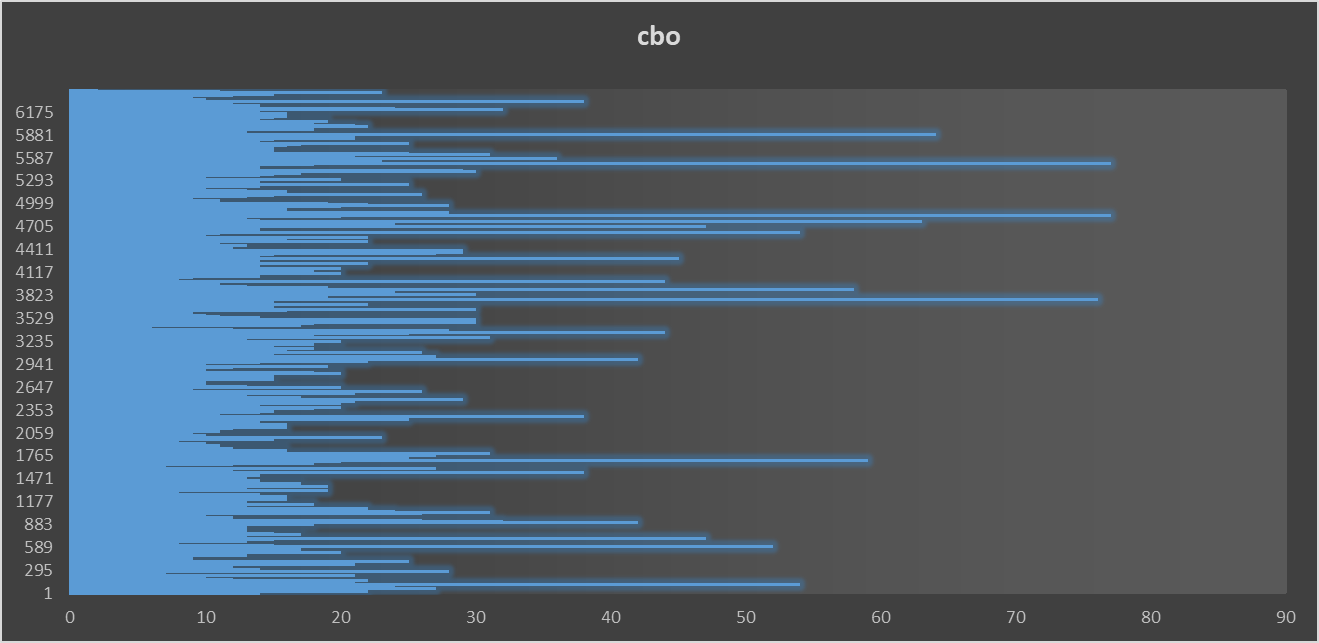
****

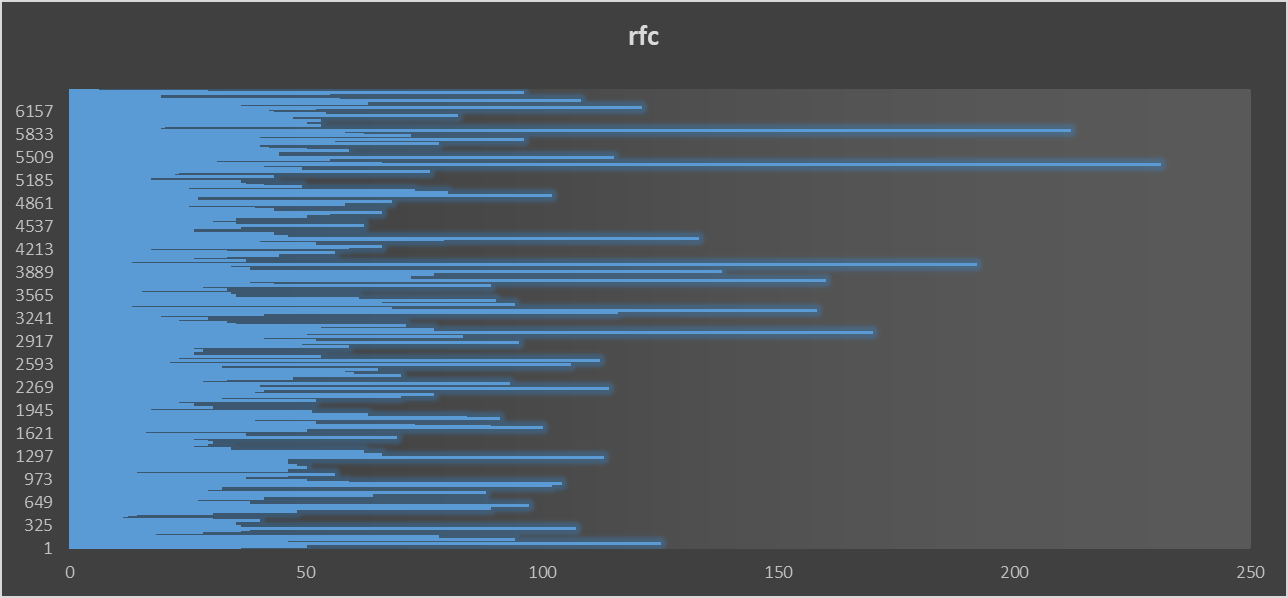
****

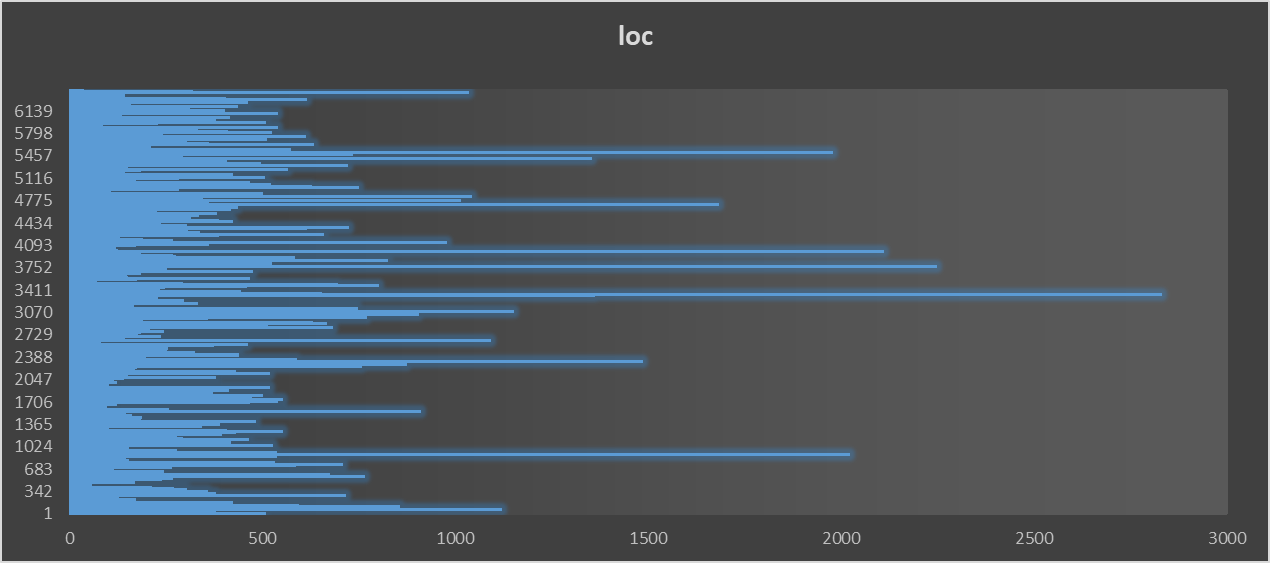
****

**Project4 : guava**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **CBO** | **RFC** | **LOC** |
| **Max** | 77 | 231 | 2828 |
| **Mode** | 0 | 0 | 5 |
| **Median** | 2 | 2 | 9 |
| **Std Deviation** | 5.318617 | 13.65527 | 117.855 |
| **Average** | 3.767064 | 6.370221 | 40.68782 |

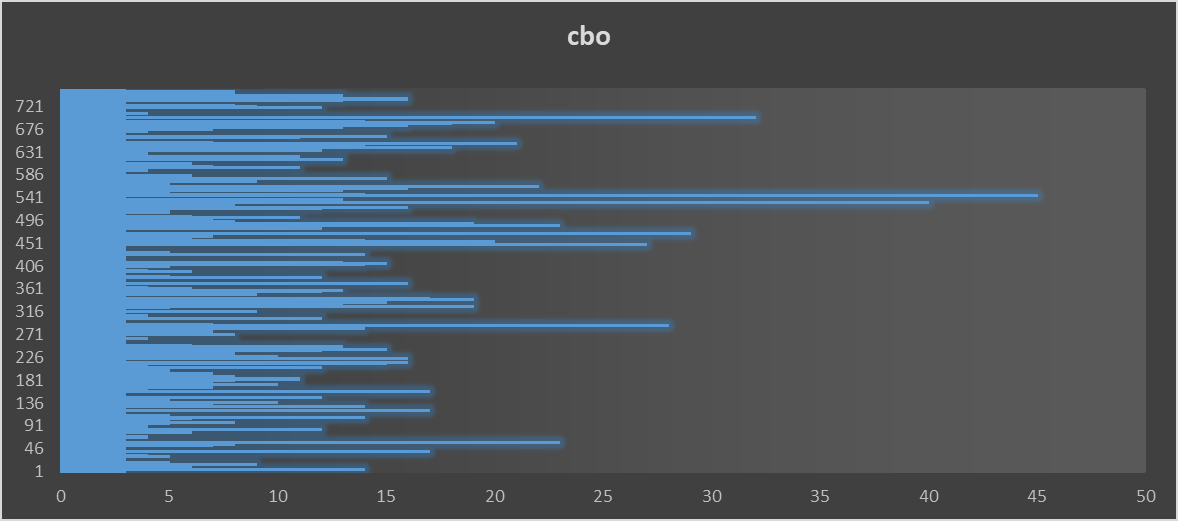
****

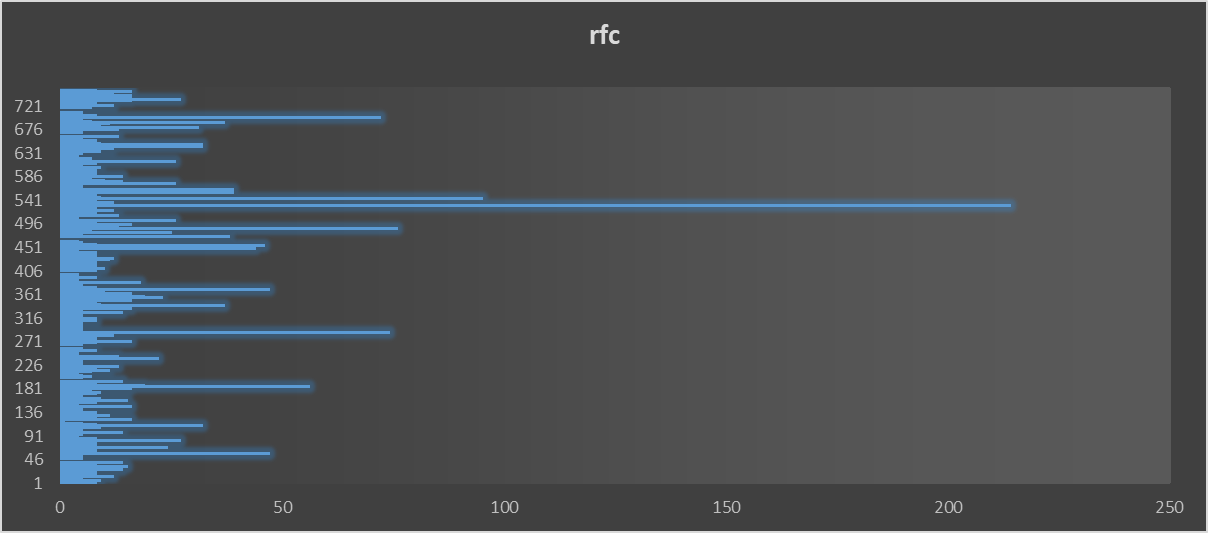
****

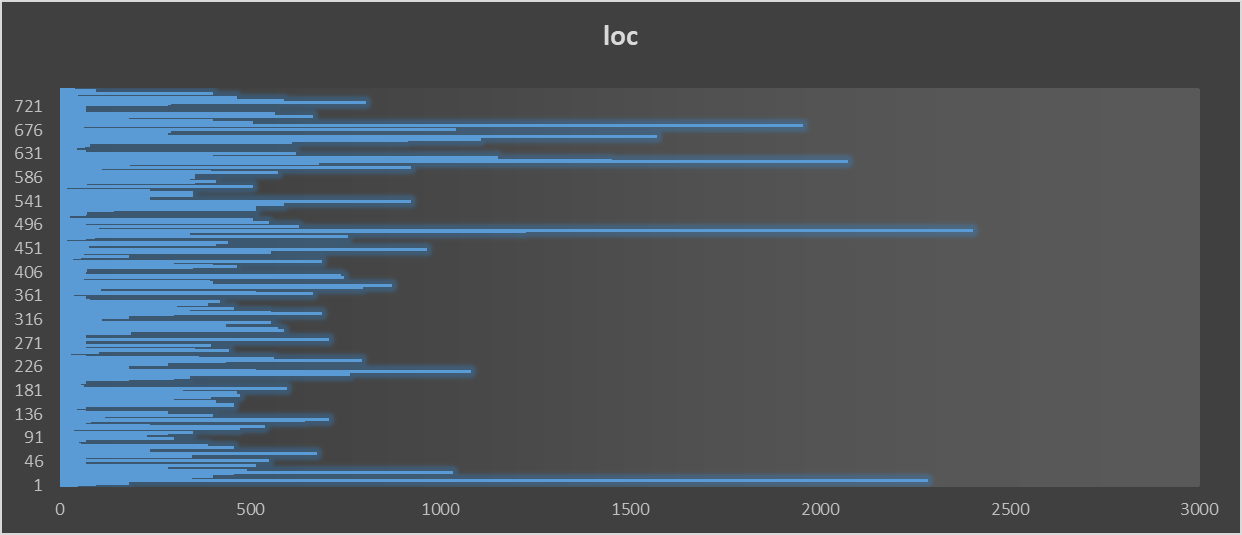
****

**Project 5: mall**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **CBO** | **RFC** | **LOC** |
| **Max** | 45 | 214 | 2400 |
| **Mode** | 3 | 0 | 5 |
| **Median** | 3 | 0 | 32 |
| **Std Deviation** | 4.946221 | 11.64802 | 261.9813 |
| **Average** | 3.912467 | 4.834218 | 133.5424 |

****

****

****

**Conclusion**

The analysis of C&K metrics can measure the set of java projects to suggest correction between class size and its maintainability. The large class can be measured by the Lines of Code (LoC), contribute higher coupling (CBO) and raise the given class(RFC) which means a positive correlation between interconnectivity and class size with other classes and number of methods within class. Elevated RFC and coupling level can impact maintainability that modification of one class needs changes to other classes and manage a large number of methods to increase the complexity. However in some cases it has a positive correlation between cohesion and size while for others the negative correlations can be observed. It means that correlation between size and cohesion is complex and relies on factors such as code structure, development methodology, and specific application domain.

The impact of size on maintainability differs among the projects. While some projects effectively handle the maintainability challenges, others face difficulties in maintaining code quality as the project size grows. This emphasizes the role of project-specific factors, like development methodologies, developer expertise, and application complexity, in the correlation between size and maintainability in Java projects. By considering these findings, developers, and organizations should approach the magnitude of java projects and consider the potential impact of maintainability. To address these challenges developers can employ the strategies like efficient code organization, adherence to design principle, and modular design that organize high cohesion and low coupling.

Further research is needed to better understand the factors that contribute to the correlations between size and maintainability in Java projects. This research should investigate how development practices, code organization, and application domain complexity impact maintainability. Additionally, it has the potential to identify effective strategies for overcoming size-related challenges in Java projects. In summary, my analysis suggests that there is a correlation between class size, coupling, and RFC in Java projects. While the relation between cohesion and size is interact and the size of on maintainability varies across the projects. The developers and organizations consider the finding to increase its development methodologies and ensure long term success.

**Reference:**

Basili, V., Caldiera, G., & Rombach, H. (n.d.). THE GOAL QUESTION METRIC APPROACH. <https://www.cs.umd.edu/users/mvz/handouts/gqm.pdf>

Elmidaoui, S., Cheikhi, L., Idri, A., & Abran, A. (2022). Towards a taxonomy of software maintainability predictors: A detailed view. In *Information Systems and Technologies: WorldCIST 2022, Volume 3* (pp. 202-210). Cham: Springer International Publishing.

Software Metrics: A Rigorous and Practical Approach, Third Edition. (n.d.). Routledge & CRC Press. Retrieved April 10, 2023, from <https://www.routledge.com/Software-Metrics-A-Rigorous-and-Practical-Approach-Third-Edition/Fenton-Bieman/p/book/9780367659028>

Eclipse JDT Language Server. (2023, April 9). GitHub. <https://github.com/eclipse/eclipse.jdt.ls>

IntelliJ Platform SDK Documentation. (2023, April 9). GitHub. <https://github.com/JetBrains/intellij-sdk-docs>

Chidamber, S. R., & Kemerer, C. F. (1994). A metrics suite for object oriented design. IEEE Transactions on Software Engineering, 20(6), 476–493. <https://doi.org/10.1109/32.295895>