

Metode Avansate de Programare, sectia Informatică 2022–2023, LABORATOR 5 - Code Complexity

DEADLINE: Săptămâna 8

Obs. Cerințele laboratorului 5 se vor realiza pe codul laboratorului 4

1. Cerința laborator 5 - Calculare Metrici de complexitate si export in fisier cvs

1. Cititi sectiunea Code Complexity – theory (vezi sectiunea 2)
2. Instalati pluginul MetricsReloaded (vezi sectiunea 3)
3. Calculare metrice de complexitate:
 - Weighted method complexity (Class metrics)
 - Cognitive complexity, Cyclomatic complexity (Method metrics)
4. Exportati rezultatul intr-un fisier cu numele grupa_NumePrenumeMetrics.csv, la urmatoarea adresa:

[Complexity](#)

2. Complexitatea codului (Code Complexity) – Suport teoretic

Introduction

Building a software system means more than delivering the functionalities within a deadline. The software development lifecycle (SDLC) is a methodology that describes the phases of building a high-quality software system (planning, requirement analysis, design, build, implementation, testing and maintenance).

One crucial aspect that impacts the testing and maintenance phases is the quality of the source code. If the code is poorly written and does not respect programming principles, it is hard to test and correct bugs that arise in the maintenance phase.

Some of the most known quality attributes are:

- **Understandability/Readability**– the code can be understood by someone who did not work on it. A well-written code can be documentation itself.
- **Maintainability** – fixing bugs and adding new functionalities are not impacted by the state of the code.
- **Consistency** – Code formatting must be consistent and follow relevant coding conventions.
- **Testability/Reliability**– The fewer bugs a software has, the higher its quality.

- **Efficiency** – High-quality code optimizes the resources available to perform the desired action.

What is code complexity?

A straightforward definition of code complexity would be how effortlessly a developer can understand a piece of code at first sight. By analyzing the code complexity of a project, we can identify the areas which need further improvements, such as simplifying the logic and removing unnecessary logic.

Why is code complexity important?

The complexity of the code directly impacts the time for fixing a bug or adding new functionality to a low-quality source code. Therefore, analyzing the code complexity can be seen as a method by which we can predict the degree of proneness to errors. Moreover, having a clean source code means that the testing coverage can be increased, thus obtaining another quality and reliable code indicator.

How to measure code complexity?

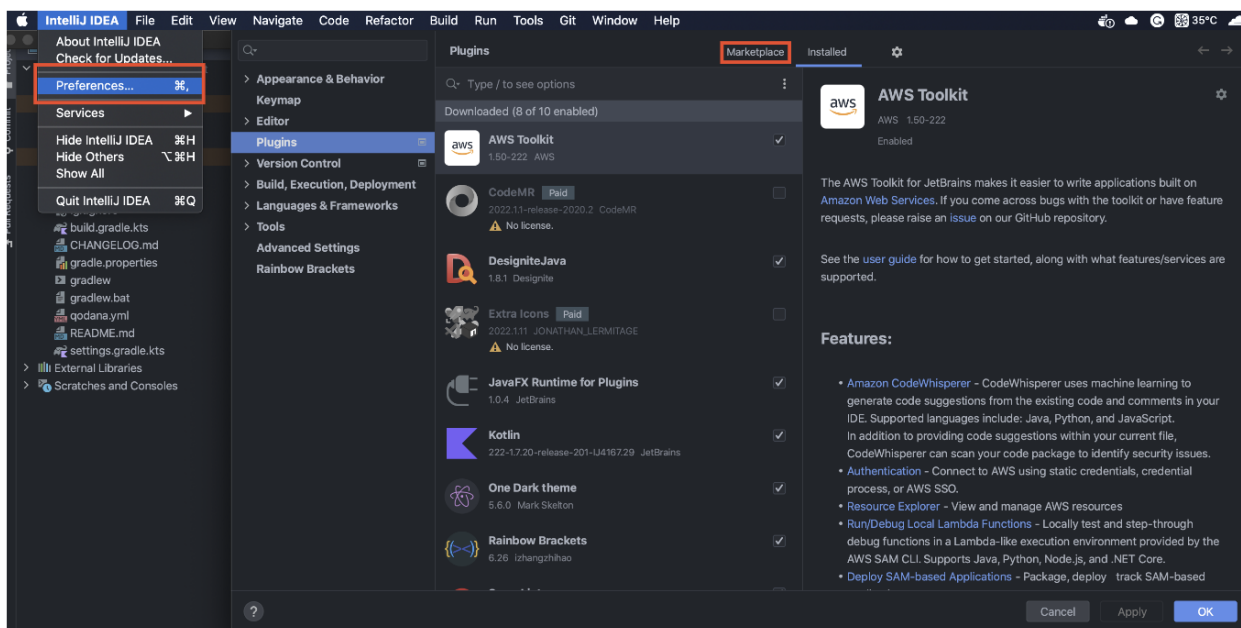
There is no doubt that to obtain code quality, we need to measure the complexity of the code, and we can do that by using software metrics.

Some of the most known code complexity software metrics are:

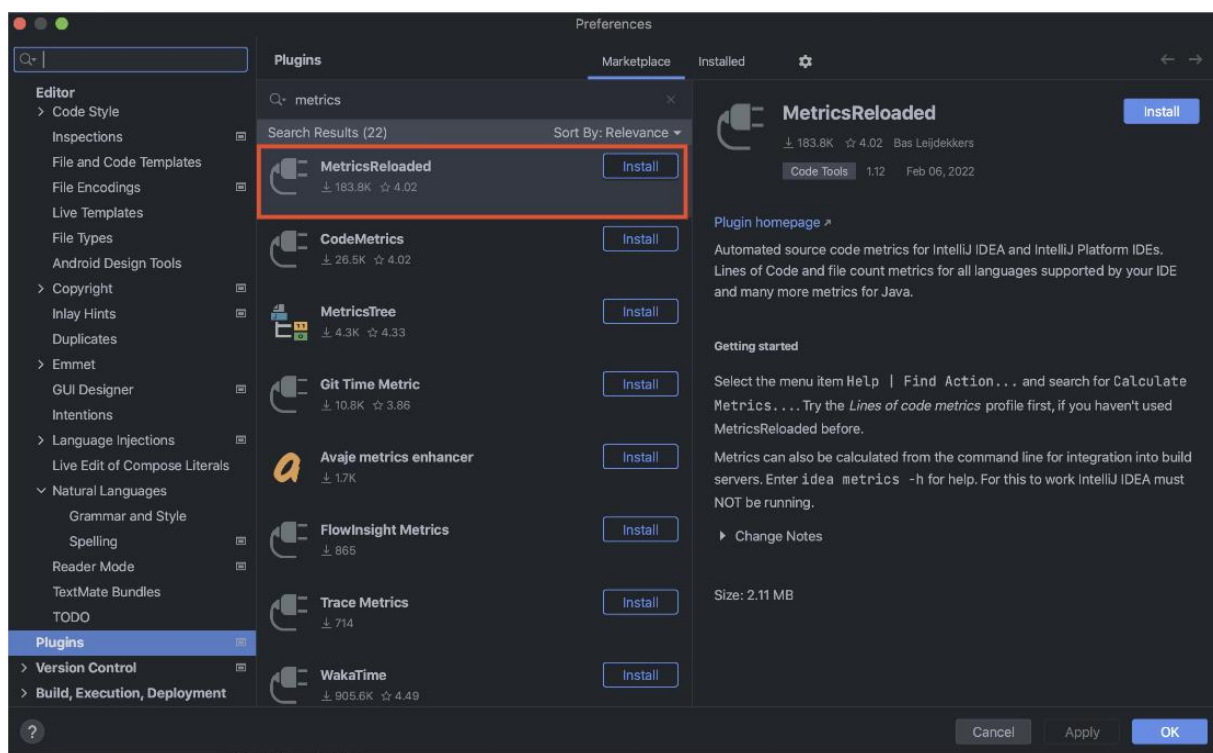
- Cyclomatic Complexity
- Source Lines of Code (SLOC)
- Halstead Volume
- Maintainability Index

3. [Instalare plugin Metrics Reloaded](#)

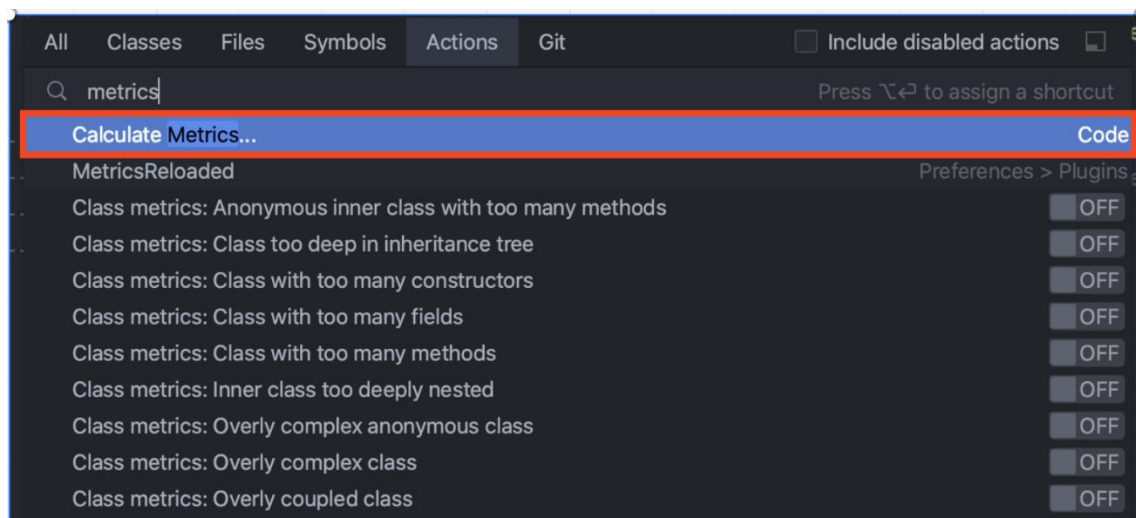
1. In **Preferences** selectati **Plugins** si mergeti la tab-ul **Marketplace**.



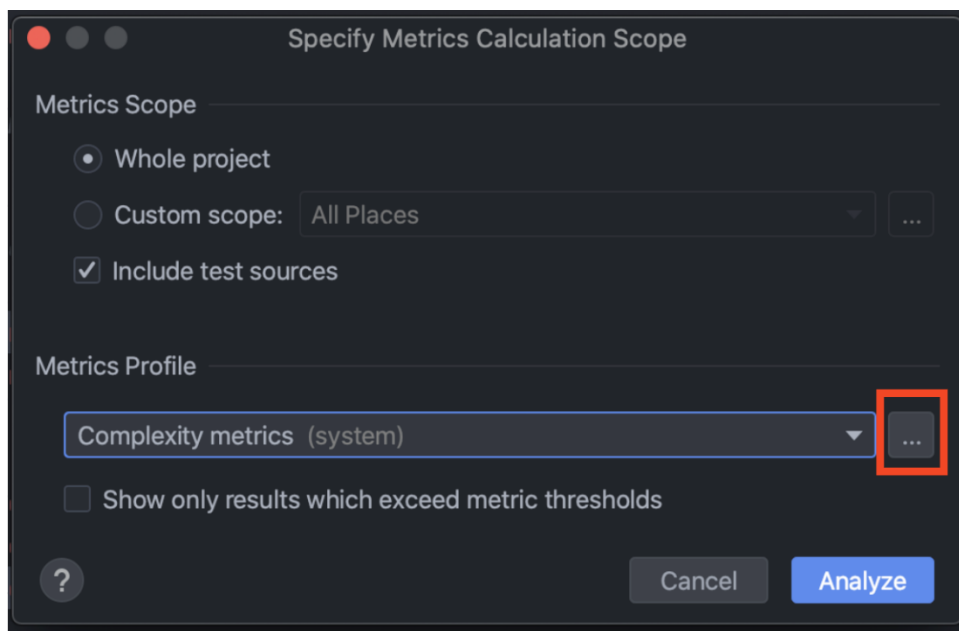
2. Cautati plugin-ul cu numele **Metrics Reloaded** si instalati-l.

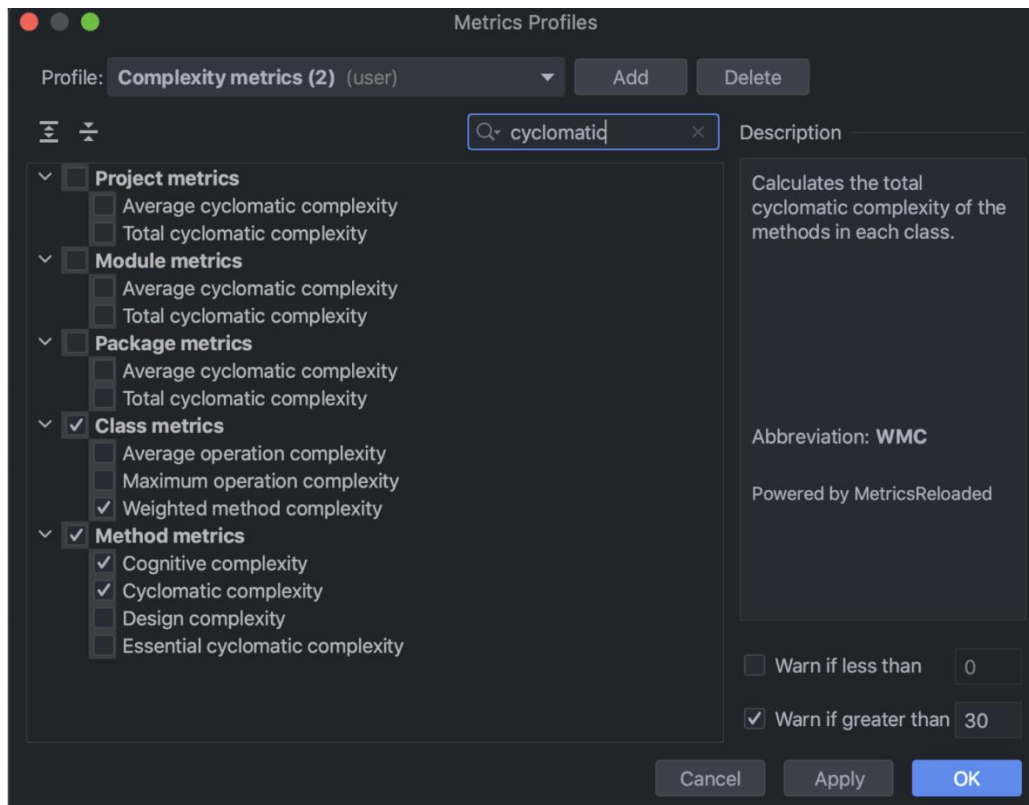


3. Pentru a testa ca plugin-ul functioneaza mergeti in meniul **Help > Find Action** si cautati “metrics”, pe urma selectati optiunea “**Calculate Metrics**”.



4. Se va deschide o fereastra de dialog unde puteti selecta lista de metrici.





5. După ce ați selectat lista de metrici apăsați **Analyze** și în partea de jos (Terminal) vor apărea rezultatele analizei.








Method metrics		Class metrics	
method		CogC	v(G)
GFG.print(Triplet[])		1	2
GFG.main(String[])		0	1
Main.main(String[])		0	1
Triplet.Triplet(int, int, int)		0	1
Triplet.compareTo(Triplet)		0	1
Triplet.toString()		0	1
Total		1	7
Average		0.17	1.17

Method metrics		Class metrics			
class		CBO	DIT	LCOM	WMC
GFG		2	1	1	3
Main		0	1	1	1
Triplet		1	1	1	3
Total					7
Average		1.00	1.00	1.00	2.33

6. Exportare rezultate

Metrics: Complexity metrics (2) for Project 'testMetricsReloaded' fro... ×

Method metrics Class metrics

method	CogC	v(G)
  ClassB.print(ClassA[])	1	2
 ClassA.ClassA(int, int, int)	0	1
 ClassA.compareTo(ClassA)	0	1
 ClassA.toString()	0	1
 ClassB.main(String[])	0	1
 Main.main(String[])	0	1
Total	1	7
Average	0.17	1.17

BOOKMARKS