

Assignment: Metrika, pregled i statička analiza

Kurs: Software Quality Introduction

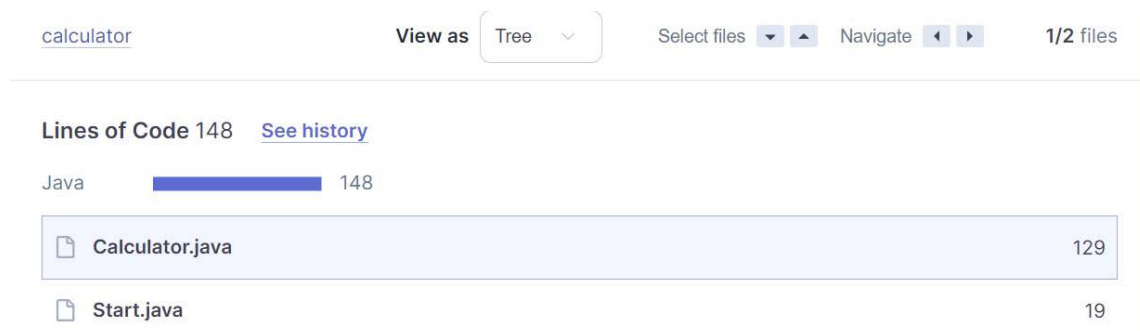
Modul: Kontrola i osiguravanje kvaliteta

Alati: **SonarQube**, **Codalyze** i **SonarLint (Visual Studio Code)**

Metrika:

1. LOC – Lines / Lines of Code – SonarQube

- Calculator.java = 188 / 129
- Start.java = 26 / 19
- Calculator.java & Start.java = 214 / 148



2. Ciklomatksa složenost fajlova (Codalyze) Calculator.java

- Calculator::Operations::Operations = 1
- Calculator::Operations::ToString = 1
- Calculator::Run = 1
- Calculator::evaluateExpression = 12
- Calculator::Calculate = 12

Function Name	Start Line	End Line	Cyclomatic Complexity (Threshold: 10)	Lines of Code (Threshold: 50)	Parameter Count (Threshold: 4)
Calculator::Operations::Operations	15	16	1	2	0
Calculator::Operations::ToString	18	20	1	3	0
Calculator::Run	24	26	1	3	1
Calculator::evaluateExpression	28	72	▲ 12	33	1
Calculator::Calculate	74	186	▲ 12	▲ 77	2

3. Ciklomatksa složenost fajlova (Codalyze) Start.java

- Start::main = 3

Function Name	Start Line	End Line	Cyclomatic Complexity (Threshold: 10)	Lines of Code (Threshold: 50)	Parameter Count (Threshold: 4)
Start::main	5	24	3	16	1

4. Ciklomatksa složenost fajlova (SonarQube) Calculator.java & Start.java

- Calculator.java = 26
- Start.java = 3
- Calculator.java & Start.java = 29

calculator

View as Tree

Select files

Navigate



1/2 files

Cyclomatic Complexity 29 [See history](#)

Calculator.java	26
Start.java	3

5. Kognitivna složenost fajlova (SonarQube) Calculator.java & Start.java

- Calculator.java = 29
- Start.java = 4
- Calculator.java & Start.java = 33

calculator	View as	Tree	Select files	Navigate	1/2 files
Cognitive Complexity 33 See history					
 Calculator.java					29
 Start.java					4

Neformalni pregled:

Programska implementacija:

Kalkulator je implementiran korišćenjem programskog jezika Java.

Struktura programa:

Sastoji se od dva fajla: **Start.java** i **Calculator.java**.

- **Start.java** predstavlja glavnu klasu za interakciju sa korisnicima.
- **Calculator.java** sadrži logiku izračunavanja matematičkih operacija.

Matematičke operacije:

Kalkulator podržava osnovne operacije: sabiranje, oduzimanje, množenje i deljenje kao i njihovu kombinaciju. Osim osnovnih operacija, kalkulator tretira -Infinity kao negativnu beskonačnost i Infinity kao pozitivnu beskonačnost. Ukoliko izraz sadrži karakter koji ne prepoznaje ili dođe do drugih grešaka prilikom izračunavanja, kalkulator će vratiti poruku "ERROR"

Interaktivna upotreba:

Korisnici interaktivno unose matematičke izraze kroz konzolu koristeći Start.java.

Petlja za unos i izračunavanje:

Unos i izračunavanje se vrše kroz beskonačnu petlju u Start.java. Petlja se završava kada korisnik unese "exit" za izlaz iz programa.

Otvoren kod i licenca:

Program je otvorenog koda i koristi MIT licencu koja omogućava slobodnu upotrebu, distribuciju i modifikaciju softvera.

Opšta Funkcionalnost:

- Program predstavlja kalkulator koji omogućava unos matematičkih izraza putem komandne linije.
- Kalkulator podržava osnovne aritmetičke operacije (sabiranje, oduzimanje, množenje i deljenje).
- Korisnici mogu uneti matematički izraz (npr. "2 + 3 * 5"), a program će ga obraditi i izračunati.
- Nakon izračunavanja rezultata, program ispisuje dobijeni rezultat na ekranu, omogućavajući korisnicima da vide rezultat unetog matematičkog izraza.

Statička analiza:

U Visual Studio Codu za statičku analizu korišćen je **Sonarlint** koji je ukazao na sledeće preporuke:

Calculator.java:

Calculator.java – In 1 – "Move this file to a named package."

Klasa "Calculator.java" nije smeštena u nijedan poseban paket (package). Ova struktura može dovesti do problema prilikom organizacije koda. Problem koji je označen kao "Move this file to a named package" sugeriše da je dobra praksa organizovati klase u određene pakete.

Calculator.java – In 4 – "Add a private constructor to hide the implicit public one."

SonarLint upozorava na to da u klasi **Calculator** postoji javni konstruktor i sugeriše da se doda privatni konstruktor. Ovaj privatni konstruktor će sprečiti implicitno stvaranje instanci klase Calculator izvan same klase. To je praksa koja pomaže u održavanju kontrole nad kreiranjem objekata.

Calculator.java – In 18 – "Rename method \"ToString\" to prevent any misunderstanding/clash with method \"toString\" defined in superclass \"java.lang.Object\"."

Calculator.java – In 18 – "Rename this method name to match the regular expression '[a-z][a-zA-Z0-9]*\$'."

Sonarlint ukazuje na potrebu preimenovanja metode **Tostring** u klasi Calculator. Ovaj problem je identifikovan kako bi se izbegao eventualni nesporazum ili konflikt sa metodom **toString** koja je već definisana u nadklasi java.lang.Object.

Sonarlint ukazuje na to da bi naziv metode trebao biti promenjen kako bi se uklopio u određeni obrazac naziva. Konkretno, preporučuje se da naziv metode počne malim slovom i sadrži samo slova (velika i mala) i brojeve.

Calculator.java – In 24 – "Rename this method name to match the regular expression '^[a-z][a-zA-Z0-9]*\$'."

Sonarlint ukazuje na to da bi naziv metode **Run** trebao biti promenjen u **run** kako bi se pridržavali konvencije da metode počinju malim slovom i sadrže samo slova (velika i mala) i brojeve.

Calculator.java – In 70 – "Immediately return this expression instead of assigning it to the temporary variable \"textResult\"."

Sonarlint ukazuje za liniju koda u kojoj se koristi privremena promenljiva **textResult**. Trebalo bi prepraviti taj deo koda kako bi se izraz odmah vratio, umesto što se dodeljuje privremenoj promenljivoj.

Calculator.java – In 74 – "Rename this method name to match the regular expression '^[a-z][a-zA-Z0-9]*\$'."

Sonarlint ukazuje da metodu **Calculate** treba preimenovati u **calculate** kako bi se ispoštovala konvencija da metode počinju malim slovom i sadrže samo slova (velika i mala) i brojeve.

Calculator.java – In 183 – "Remove this redundant jump."

Sonarlint ukazuje da postoji nepotrebna instrukcija **return**; nakon rekurzivnog poziva metode **Calculate**, budući da će sam rekurzivni poziv već završiti izvršavanje metode.

Start.java:

Start.java – In 1 – "Move this file to a named package."

SonarLint sugeriše da fajl treba premestiti u paket.

Start.java – In 6 – "Rename this local variable to match the regular expression '^[a-z][a-zA-Z0-9]*\$'."

Ova poruka sugeriše da se preimenuje lokalna promenljiva **String Expression** tako da odgovara regularnom izrazu **^[a-z][a-zA-Z0-9]*\$**, na primer **String expression** ili **String inputExpression**. Naziv promenljive treba da počne malim slovom i može sadržavati slova (mala i velika) ili cifre.

Start.java – In 8 – In 19 – "Replace this use of System.out by a logger."

SonarLint sugeriše da se zameni direktno korišćenje **System.out** sa mehanizmom za zapisivanje logova (logger). Loggeri omogućavaju bolju kontrolu nad ispisivanjem poruka i pružaju različite nivoe logovanja.