Cpt_S 422 Report

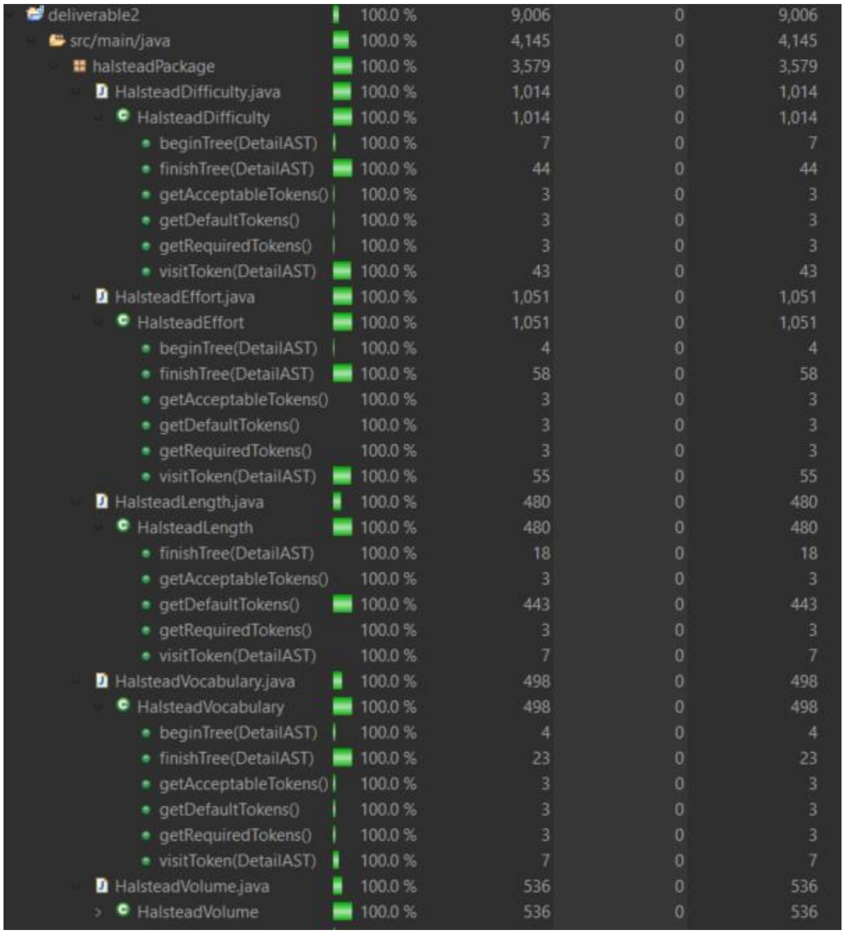
This report will explain the test cases used in all of the classes involved with this project. There will be 2 sections for this report. Halstead checks, and other checks. Since there were two categories of code there will be two categories of tests.

Halstead:

Each test on each class has tested every single method within the class. Each one utilizes mock and spies to test functionality of each method even if there's no information passed into the variables. This is an example of black box testing since we don't know the full functionality of the Halstead checks in real time.

Classes tested:

- Length
- Vocabulary
- Volume
- Difficulty
- Effort



deliverable2	100.0 %	9,006	0	9,006
src/main/java	100.0 %	4,145	0	4,145
halsteadPackage	100.0 %	3,579	0	3,579
HalsteadDifficulty.java	100.0 %	1,014	0	1,014
HalsteadDifficulty	100.0 %	1,014	0	1,014
beginTree(DetailAST)	100.0 %	7	0	7
finishTree(DetailAST)	100.0 %	44	0	44
getAcceptableTokens()	100.0 %	3	0	3
getDefaultTokens()	100.0 %	3	0	3
getRequiredTokens()	100.0 %	3	0	3
visitToken(DetailAST)	100.0 %	43	0	43
HalsteadEffort.java	100.0 %	1,051	0	1,051
HalsteadEffort	100.0 %	1,051	0	1,051
beginTree(DetailAST)	100.0 %	4	0	4
finishTree(DetailAST)	100.0 %	58	0	58
getAcceptableTokens()	100.0 %	3	0	3
getDefaultTokens()	100.0 %	3	0	3
getRequiredTokens()	100.0 %	3	0	3
visitToken(DetailAST)	100.0 %	55	0	55
HalsteadLength.java	100.0 %	480	0	480
HalsteadLength	100.0 %	480	0	480
finishTree(DetailAST)	100.0 %	18	0	18
getAcceptableTokens()	100.0 %	3	0	3
getDefaultTokens()	100.0 %	443	0	443
getRequiredTokens()	100.0 %	3	0	3
visitToken(DetailAST)	100.0 %	7	0	7
HalsteadVocabulary.java	100.0 %	498	0	498
HalsteadVocabulary	100.0 %	498	0	498
beginTree(DetailAST)	100.0 %	4	0	4
finishTree(DetailAST)	100.0 %	23	0	23
getAcceptableTokens()	100.0 %	3	0	3
getDefaultTokens()	100.0 %	3	0	3
getRequiredTokens()	100.0 %	3	0	3
visitToken(DetailAST)	100.0 %	7	0	7
HalsteadVolume.java	100.0 %	536	0	536
HalsteadVolume	100.0 %	536	0	536

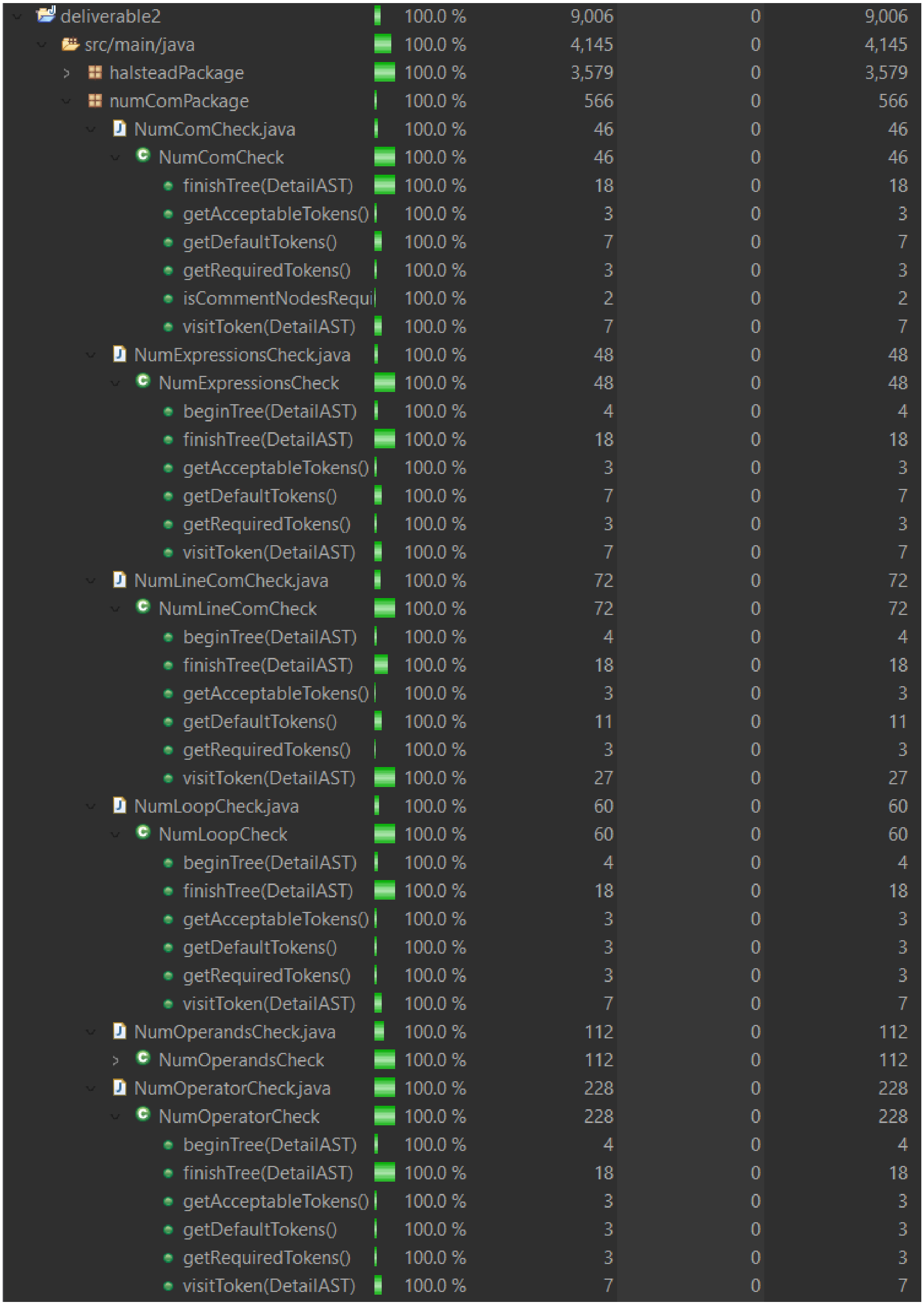
Others:

Each test for each of the classes within this category covers every function/method. This makes for 100% coverage. The tests here utilize mock and spy because we are using a testing method known as black box where we don’t know the full functionality of the code when we run it beforehand.

Classes tested:

- NumOperators
- NumOperands
- NumExpression
- NumLoops
- NumComments
- NumLineComments

There is a screenshot on the next page covering all the methods that were tested.



deliverable2	100.0 %	9,006	0	9,006
src/main/java	100.0 %	4,145	0	4,145
halsteadPackage	100.0 %	3,579	0	3,579
numComPackage	100.0 %	566	0	566
NumComCheck.java	100.0 %	46	0	46
NumComCheck	100.0 %	46	0	46
finishTree(DetailAST)	100.0 %	18	0	18
getAcceptableTokens()	100.0 %	3	0	3
getDefaultTokens()	100.0 %	7	0	7
getRequiredTokens()	100.0 %	3	0	3
isCommentNodesRequi	100.0 %	2	0	2
visitToken(DetailAST)	100.0 %	7	0	7
NumExpressionsCheck.java	100.0 %	48	0	48
NumExpressionsCheck	100.0 %	48	0	48
beginTree(DetailAST)	100.0 %	4	0	4
finishTree(DetailAST)	100.0 %	18	0	18
getAcceptableTokens()	100.0 %	3	0	3
getDefaultTokens()	100.0 %	7	0	7
getRequiredTokens()	100.0 %	3	0	3
visitToken(DetailAST)	100.0 %	7	0	7
NumLineComCheck.java	100.0 %	72	0	72
NumLineComCheck	100.0 %	72	0	72
beginTree(DetailAST)	100.0 %	4	0	4
finishTree(DetailAST)	100.0 %	18	0	18
getAcceptableTokens()	100.0 %	3	0	3
getDefaultTokens()	100.0 %	11	0	11
getRequiredTokens()	100.0 %	3	0	3
visitToken(DetailAST)	100.0 %	27	0	27
NumLoopCheck.java	100.0 %	60	0	60
NumLoopCheck	100.0 %	60	0	60
beginTree(DetailAST)	100.0 %	4	0	4
finishTree(DetailAST)	100.0 %	18	0	18
getAcceptableTokens()	100.0 %	3	0	3
getDefaultTokens()	100.0 %	3	0	3
getRequiredTokens()	100.0 %	3	0	3
visitToken(DetailAST)	100.0 %	7	0	7
NumOperandsCheck.java	100.0 %	112	0	112
NumOperandsCheck	100.0 %	112	0	112
NumOperatorCheck.java	100.0 %	228	0	228
NumOperatorCheck	100.0 %	228	0	228
beginTree(DetailAST)	100.0 %	4	0	4
finishTree(DetailAST)	100.0 %	18	0	18
getAcceptableTokens()	100.0 %	3	0	3
getDefaultTokens()	100.0 %	3	0	3
getRequiredTokens()	100.0 %	3	0	3
visitToken(DetailAST)	100.0 %	7	0	7