# halstead-undip

*by* Iwan Binanto

**Submission date:** 24-Jul-2018 05:02PM (UTC+0700)

**Submission ID:** 862006451

**File name:** Halstead-UNDIP1.pdf (1.45M)

**Word count:** 2729

**Character count:** 14895

# Halstead Metric for Quality Measurement of Various Version of Statcato

[1,2] Iwan Binanto
[1]Informatics Department,
Faculty of Scence and Engineering
Sanata Dharma University
Yogyakarta, Indonesia 55002

[2]Computer Science Department, BINUS
Graduate Program – Doctor of Computer
Science
Bina Nusantara University
Jakarta, Indonesia 11480
iwan@usd.ac.id

[1]Harco Leslie Hendric Spits
Warnars, [2]Bahtiar Saleh Abbas,
Computer Science Department, BINUS
Graduate Program – Doctor of
Computer Science
Bina Nusantara University
Jakarta, Indonesia 11480
spits.hendric@binus.ac.id[1],
bahtiars@binus.edu[2]

[1,2]Nesti Fronika Sianipar
[1]Food Technology Department,
Faculty of Engineering,
Bina Nusantara University,
Jakarta, Indonesia 11480

[2]Research Interest Group
Biotechnology,
Bina Nusantara University,
Jakarta, Indonesia 11480
nsianipar@binus.edu

***Abstract*—Halstead metric is one of software quality measurement technique. Some studies call it as Primitive and/or Classic Software Metric, although some researchers find vagueness of it. This paper utilizing Halstead metric to measuring the quality of various versions of Statcato software. Measurements are performed in each class of each version of Statcato software. Python programming used to facilitate this. This paper find that in terms of Halstead metric, this software seems stable during its lifecycle. So, the quality of the newer versions of Statcato are better in terms of stability of Halstead metrics than the earlier version, although its feature is increasing.**

***Keywords*—python programming; halstead metric; statcato; quality measurement**

## I. Introduction

There are many published papers on the topic of software metrics with various metrics [1]. The widely used measurement is Halstead Metric [2], Dand and Vasishtha classified it as Primitive Software Metric [1] and Moreno-Leon et al. called it traditional software engineering metric [3]. Many papers published using this metric [3]–[10]. Nevertheless, some researchers find vagueness measuring with Halstead Metrics and consider that Maurice Halstead did not understand the theory of measurement [2], [5].

Halstead Metrics can be calculated manually or using the software. There is free software written in Java by Ramasubramanian namely *HalsteadMetrics.jar and HalsteadMetricsCMD.jar* that efficiently help to calculate Halstead Metrics [11]. Our study uses one of this software which is *HalsteadMetricsCMD.jar* because of it faster than the other. In our opinion, this software is beneficial in research of measurement software, especially in Halstead Metrics. That software will generate .html or .pdf files as output files.

We need a .html instead of .pdf files for further processing in our research. The problem arises since the .html files generated by *HalsteadMetrics.jar or HalsteadMetricsCMD.jar* is more than 1000 files. It will take a long time to retrieve some of the data in the .html files manually, such as blocking and copying a part of data and pasting it into, example, Microsoft Excel, for further processing. Consequently, we need a tool to do it automatically.

At this time, many famous and important applications are open source software. This kind of software used by many universities and business companies for their solutions. Open source software is developed with different management styles from which industry, quality, and reliability of the code should be investigated [12].

One scientific application that is open source software is Statcato. It is a Java-based application which is naturally an Object-Oriented Software. Once any machine whether runs Windows, Mac, Linux, or UNIX has Java Runtime Environment (JRE), Statcato can be run [13].

The aim of this study measures the quality of various version of Statcato during its lifecycle using Halstead Metric. Metrics value are calculated on multiple versions that are used as a basis for evaluating the quality of Statcato software. To do this, we also developing tools to fit our needs that will retrieve a part of data from the output of *HalsteadMetricsCMD.jar* which are .html files and store it as .csv files. Statcato had been studied by Binanto et al. using CK Metric, with the result that its quality has improved over its life cycle, despite the features increase [14].

## II. Related Works

Halstead measurement remains a mystery issue. The nature of these steps is not well understood, and there is still a scale confusion (type). Halstead measurements are used by many software, but the usefulness and meaning of the number of steps are not evident [2]. Qutaish and Abran also agree with this, that Halstead measurements do not meet the criteria of crucial design for analysis in engineering and the physical sciences [5]. Nevertheless, Moreno-Leon et al. argue that Halstead measurement is globally recognized as a valid measurement for the complexity of a software system. They successfully used it to measure complexity Dr. Scratch software and make a comparison between itself, McCabe's Cyclomatic Complexity, and the computational thinking score provided by Dr. Scratch software [3].

Meulen and Revilla revisited Halstead measurement to find a correlation between internal software metric and software dependability, and one of their conclusion is a robust correlation found between Line of Code and Halstead Volume [4]. Cornwell said there are three most important metrics measured are Program Length, Volume, and Effort. Two of them correlated with maintainability, which is Volume and Effort [7]. Sehgal and Mehrotra predict a program's failure before the testing phase using one of Halstead's metrics, which is Program Volume. It is used to calculate the number of faults in the program [8].

Parade et al. have developed a software for calculating Halstead metrics using python for automation to determine complexity software quality. They argue that the data flow of the programs will be derived in Halstead ~~metrics and~~ returns parameters like effort ~~and difficulty~~ level which are very helpful [6].

Spinellis created a tool based on Java to fit his needs and discovered something important that writing stand-alone tools becoming a forgotten art [13]. Python is a high-level programming language that allows programmers to work faster toward problem-solving effectively. It was designed as a general-purpose programming language and become the valid choice of scientists for helping their research by developing scientific programs. The use of Python is multiplying because of the dedicated work of many contributors over the years. Python has many libraries to express many complex algorithmic questions with clarity and efficiency [15]. Many published papers related to python and software metric. Misra and Cafer work on evaluating Python, Java, and C++ which are an object oriented programming language and conclude that Python is better than two others on their case study [16]. Thirumalai, et al. used a program written in Python as a subject for their research to evaluate Halstead Metric [10]. Shudrak and Zolotarev by using Python, found that Halstead's B metrics are the best and most appropriate metric for finding bugs [17].

## III. Halstead Metric: An Overview

The software is a set of commands/computer programs that are a group of tokens. Tokens can be classified as operators or operands. Therefore, "a program can be considered a sequence of related operators and operands." Calculation of Halstead metric is based on the total number of tokens in a given source code program. First, the source code program divided into tokens and then all counted and classified as operators or operands [6]. The program is not a program without operators and operands.

The goal of the Halstead measurement is to measure certain qualities, for example, vocabulary, volume, level, problem program deployment and required programming time [9]. Halstead argues that algorithms have measurable characteristics analogous to physical law. In the given program, it counts the number of unique or different operators (n1), single or different operands (n2), total use of all operators (N1) and overall use of all operands (N2) [1]. So, the Halstead measurement based on:

n1 = number of unique or distinct operators.
n2 = number of unique or distinct operands.
N1 = total number of occurrences of operators.
N2 = total number of occurrences of operands.

Based on these data, Halstead measurements define some metrics, which are [11]:

### A. Program Length

This is the total number of occurrences of operators and operands (repetition also considered) which is called the length of the program. The formula is:

$$program\ length = N1 + N2 \quad (1)$$

### B. Program Vocabulary

This is the sum of the number of unique or distinct operators and the number of unique or distinct operands in the program. The formula is:

$$programVocabulary = n1 + n2 \quad (2)$$

### C. Estimated Length

This is called a calculated program length which estimated the program overall. The formula is:

$$estimated\ length = (n1 * log_2(n1)) + (n2 * log_2(n2)) \quad (3)$$

### D. Purity Ratio

This is a ratio of estimated program length with a calculation of program length. The formula is:

$$purityRatio = \frac{estimatedLength}{programLength} \quad (4)$$

### E. Program Volume

This is the number of bits required to represent the program [3] or alternative measure for program size [18]. The formula is:

$$volume = programLength * log_2\ (programVocabulary) \quad (5)$$

### F. The difficulty of The Program

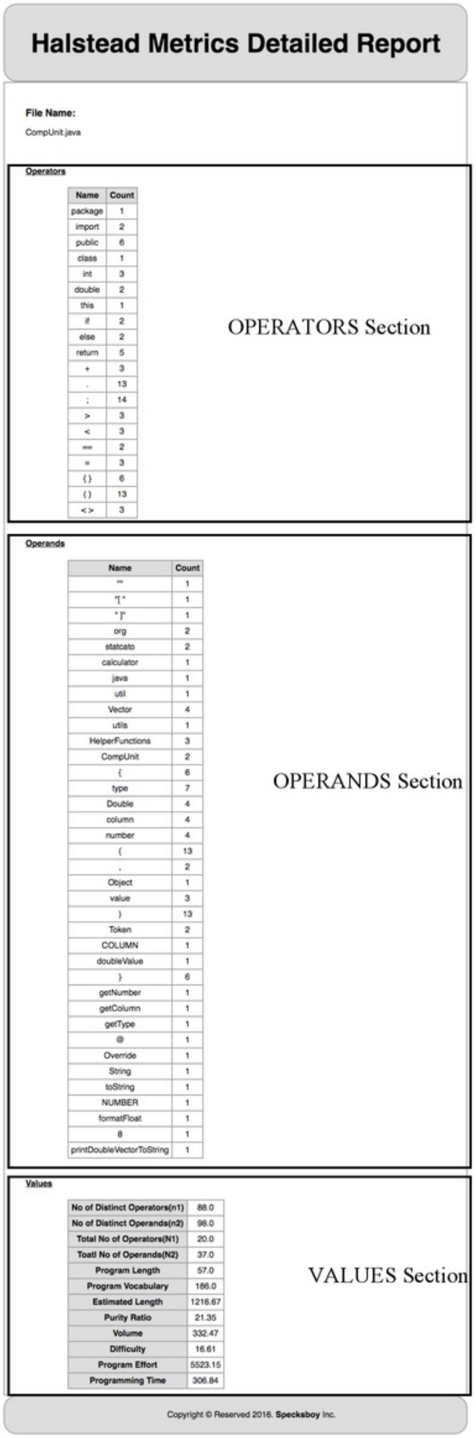
This is used to compare various applications of the same algorithm. The longer the implementation, the greater the difficulty. The formula is:

$$difficulty = \left(\frac{n1}{2}\right) * \left(\frac{N2}{n2}\right) \quad (6)$$

### G. Program Effort

This is the effort required to understand or create a program. The formula is:

$$programEffort = difficulty * volume \quad (7)$$



**Halstead Metrics Detailed Report**

**File Name:**

CompUnit.java

**Operators**

| Name | Count |
|---|---|
| package | 1 |
| import | 2 |
| public | 6 |
| class | 1 |
| int | 3 |
| double | 2 |
| this | 1 |
| if | 2 |
| else | 2 |
| return | 5 |
| + | 3 |
| . | 13 |
| ; | 14 |
| > | 3 |
| < | 3 |
| == | 2 |
| = | 3 |
| { } | 6 |
| ( ) | 13 |
| < > | 3 |

OPERATORS Section

**Operands**

| Name | Count |
|---|---|
| "" | 1 |
| "[ " | 1 |
| " ]" | 1 |
| org | 2 |
| statcato | 2 |
| calculator | 1 |
| java | 1 |
| util | 1 |
| Vector | 4 |
| utils | 1 |
| HelperFunctions | 3 |
| CompUnit | 2 |
| { | 6 |
| type | 7 |
| Double | 4 |
| column | 4 |
| number | 4 |
| ( | 13 |
| , | 2 |
| Object | 1 |
| value | 3 |
| ) | 13 |
| Token | 2 |
| COLUMN | 1 |
| doubleValue | 1 |
| } | 6 |
| getNumber | 1 |
| getColumn | 1 |
| getType | 1 |
| @ | 1 |
| Override | 1 |
| String | 1 |
| toString | 1 |
| NUMBER | 1 |
| formatFloat | 1 |
| 8 | 1 |
| printDoubleVectorToString | 1 |

OPERANDS Section

**Values**

| | |
|---|---|
| No of Distinct Operators(n1) | 88.0 |
| No of Distinct Operands(n2) | 98.0 |
| Total No of Operators(N1) | 20.0 |
| Toatl No of Operands(N2) | 37.0 |
| Program Length | 57.0 |
| Program Vocabulary | 186.0 |
| Estimated Length | 1216.67 |
| Purity Ratio | 21.35 |
| Volume | 332.47 |
| Difficulty | 16.61 |
| Program Effort | 5523.15 |
| Programming Time | 306.84 |

VALUES Section

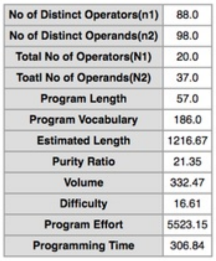
Copyright © Reserved 2016. **Specksboy** Inc.

Fig. 1. The .html page structure as output from HalsteadMetricsCMD.jar

### *H. Programming Time*

This is the time required to build a program. It can be calculated as:

$$programmingTime = \frac{programEffort}{18} \quad (8)$$



| | |
|---|---|
| No of Distinct Operators(n1) | 88.0 |
| No of Distinct Operands(n2) | 98.0 |
| Total No of Operators(N1) | 20.0 |
| Toatl No of Operands(N2) | 37.0 |
| Program Length | 57.0 |
| Program Vocabulary | 186.0 |
| Estimated Length | 1216.67 |
| Purity Ratio | 21.35 |
| Volume | 332.47 |
| Difficulty | 16.61 |
| Program Effort | 5523.15 |
| Programming Time | 306.84 |

Fig. 2. The results of data pruning in the .html file

## IV. METHOD AND DISCUSSIONS

There are 3 (three) sections of .html display structure generated by *HalsteadMetricsCMD.jar* as in Fig. 1. We need to retrieve the value in VALUES Section only because in this section halstead metrics reside. Consequently, we will prune two previous sections, which are OPERATORS and OPERANDS, then the result converted into one .csv file for data in the tabulation, which will make easier for further processing. The pruning .html file illustrated as in Fig. 2.

For this job, we use Python language for programming because it is a high-level language which is easy to solve technical problems like in this paper and have many libraries ready to use [15]. It is also freely available in source or binary form from the Python site and can be distributed for free [19]. Furthermore, Sanner stated that dynamic, easy to simplify, and easy-to-use in Python make this language an excellent choice for creating powerful modern software [20].

One .html file produce by *HalsteadMetricsCMD.jar* was representing one file or class of each one version Statcato software. There are many files or classes in each version of Statcato software. For this study, we retrieve information one by one per file or class in Statcato's packages. This .html files contains information of Halstead metrics which are Program Length, Program Vocabulary, Estimated Length, Purity Ratio, Program Volume, Difficulty of Program, Program Effort, and Programming Time.

The total number of these .html files more than one thousand and we must combine these files to be one .csv file only. In other words, the VALUES Sections from thousand file .html. We especially numbers just should be merged into one .csv file. We call this prunning .html file to .csv as illustrated in Fig. 2 and done by Python programming. The data that will be combined are all the second column of the .csv file; this is where the required data resides.

The combining process illustrated in Fig. 3 where all of .csv file produce by Python programming combine from early version to last version of Statcato into one merged .csv file only as illustrated in Fig. 4. To do this we write a Python code as in Fig. 5.

Since all information of files or classes gathered in one version of package Statcato, then combine into one .csv file, and the mean per metric calculated. We do this for all of Statcato's version. So, at the end process, we only have one .csv file that is representing the mean per metric of one version of Statcato software as in Table 1. From this table, we plotting metrics of Halstead as in Fig. 6.

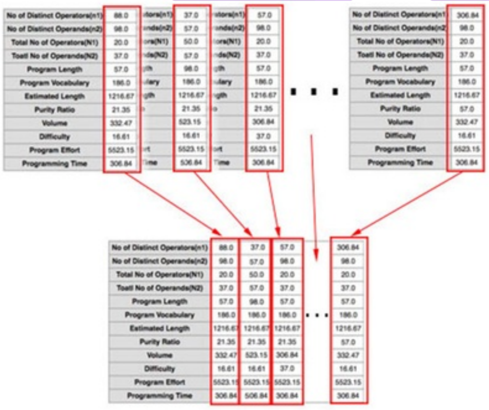
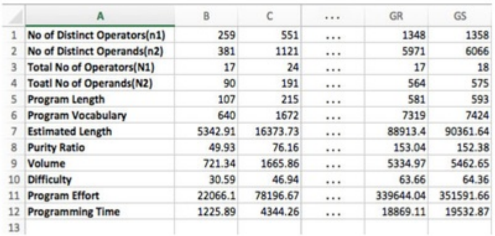
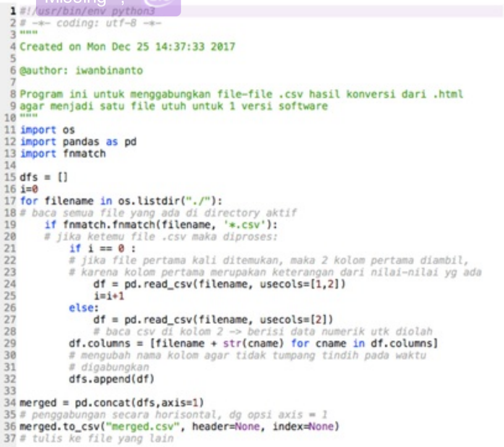


Fig. 3. Illustration how the second columns of many tables, merge into one table



| | A | B | C | ... | GR | GS |
|---|---|---|---|---|---|---|
| 1 | No of Distinct Operators(n1) | 259 | 551 | ... | 1348 | 1358 |
| 2 | No of Distinct Operands(n2) | 381 | 1121 | ... | 5971 | 6066 |
| 3 | Total No of Operators(N1) | 17 | 24 | ... | 17 | 18 |
| 4 | Toatl No of Operands(N2) | 90 | 191 | ... | 564 | 575 |
| 5 | Program Length | 107 | 215 | ... | 581 | 593 |
| 6 | Program Vocabulary | 640 | 1672 | ... | 7319 | 7424 |
| 7 | Estimated Length | 5342.91 | 16373.73 | ... | 88913.4 | 90361.64 |
| 8 | Purity Ratio | 49.93 | 76.16 | ... | 153.04 | 152.38 |
| 9 | Volume | 721.34 | 1665.86 | ... | 5334.97 | 5462.65 |
| 10 | Difficulty | 30.59 | 46.94 | ... | 63.66 | 64.36 |
| 11 | Program Effort | 22066.1 | 78196.67 | ... | 339644.04 | 351591.66 |
| 12 | Programming Time | 1225.89 | 4344.26 | ... | 18869.11 | 19532.87 |
| 13 | | | | | | |

Fig. 4 The result of merging multiple .csv files into one



```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon Dec 25 14:37:33 2017

@author: iwanbinanto

Program ini untuk menggabungkan file-file .csv hasil konversi dari .html
agar menjadi satu file utuh untuk 1 versi software
"""
import os
import pandas as pd
import fnmatch

dfs = []
i=0
for filename in os.listdir("./"):
# baca semua file yang ada di directory aktif
    if fnmatch.fnmatch(filename, '*.csv'):
    # jika ketemu file .csv maka diproses:
        if i == 0 :
        # jika file pertama kali ditemukan, maka 2 kolom pertama diambil,
        # karena kolom pertama merupakan keterangan dari nilai-nilai yg ada
            df = pd.read_csv(filename, usecols=[1,2])
            i=i+1
        else:
            df = pd.read_csv(filename, usecols=[2])
            # baca csv di kolom 2 -> berisi data numerik utk diolah
        df.columns = [filename + str(cname) for cname in df.columns]
        # mengubah nama kolom agar tidak tumpang tindih pada waktu
        # digabungkan
        dfs.append(df)

merged = pd.concat(dfs,axis=1)
# penggabungan secara horisontal, dg opsi axis = 1
merged.to_csv("merged.csv", header=None, index=None)
# tulis ke file yang lain
```

Fig. 5. Pythons' code for combining all generated .csv files

TABLE I. THE MEAN OF HALSTEAD METRICS OF VARIOUS STATCATO'S VERSION

| | Version of Statcato | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **0.94** | **0.95** | **0.96** | **0.97** | **0.98** | **0.99** | **0.910** | **0.911** | **0.912** | **1.0** | **1.01** | **1.02** |
| **No of Distinct Operators (n1)** | 2,787 | 2,787 | 1,704 | 1,673 | 1,684 | 1,692 | 1,713 | 1,713 | 1,720 | 1,721 | 1,721 | 1,748 |
| **No of Distinct Operands (n2)** | 11,286 | 11,287 | 5,735 | 5,615 | 5,676 | 5,704 | 5,811 | 5,814 | 5,834 | 5,839 | 5,840 | 5,985 |
| **Total No of Operators (N1)** | 28 | 28 | 27 | 27 | 27 | 27 | 27 | 27 | 27 | 27 | 27 | 27 |
| **Total No of Operands (N2)** | 379 | 379 | 305 | 300 | 301 | 304 | 304 | 302 | 303 | 303 | 303 | 312 |
| **Program Length** | 407 | 407 | 332 | 326 | 328 | 330 | 331 | 329 | 330 | 330 | 330 | 340 |
| **Program Vocabulary** | 14,073 | 14,074 | 7,439 | 7,288 | 7,360 | 7,396 | 7,523 | 7,527 | 7,554 | 7,560 | 7,561 | 7,733 |
| **Estimated Length** | 196,916 | 196,935 | 94,257 | 92,203 | 93,170 | 93,610 | 95,438 | 95,438 | 95,828 | 95,907 | 95,918 | 98,176 |
| **Purity Ratio** | 339 | 339 | 236 | 234 | 236 | 236 | 241 | 242 | 242 | 242 | 242 | 245 |
| **Volume** | 3,647 | 3,648 | 2,870 | 2,815 | 2,828 | 2,851 | 2,856 | 2,829 | 2,836 | 2,839 | 2,842 | 2,935 |
| **Difficulty** | 56 | 56 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 | 50 |
| **Program Effort** | 283,236 | 283,347 | 214,699 | 209,478 | 210,031 | 211,678 | 211,418 | 204,122 | 50 | 205,392 | 205,873 | 212,575 |
| **Programming Time** | 15,735 | 15,741 | 11,928 | 11,638 | 11,668 | 11,760 | 11,745 | 11,340 | 11,389 | 11,411 | 11,437 | 11,810 |

## V. Conclusion

In Figure 6, we can see that the previous version, the number of operators and operands are higher than the newer versions. But then, the higher the version of Statcato, there are not many changes in the number of operators and operands. Because other metrics depend on the number of operators and operands as set out in formulas (1) - (7), this does not affect other metrics, so they do not change much. We can say that in terms of Halstead metric, this software seems stable during its lifecycle.

We conclude that the quality of the newer versions are better in terms of stability of Halstead metrics than the earlier version of Statcato, although its feature is increasing. Accordingly, Statcato software has a good condition and excellent development to continue improving during its lifecycle, although its feature is expanding.

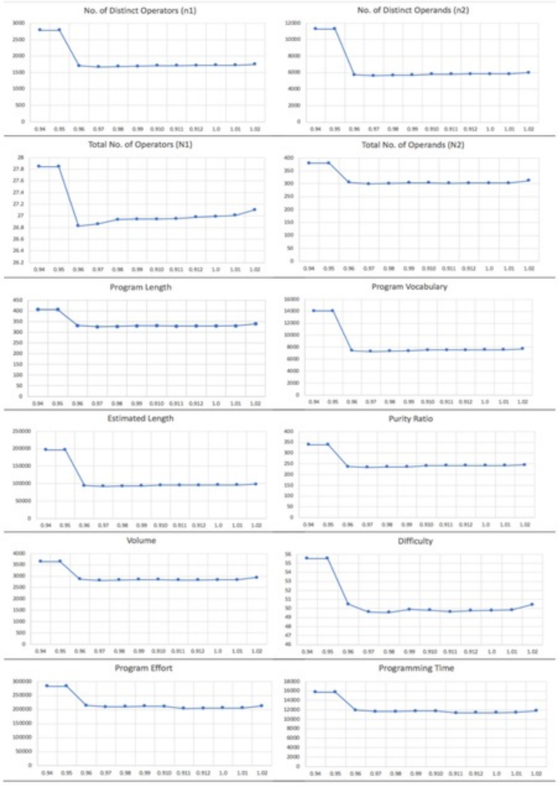


Fig 6. Halstead Metric s graphic of all Statcato's version

## References

[1] Z. G. Dand and H. Vasishtha, "Analysis and Evaluation of Quality Metrics in Software Engineering," *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 4, no. 4, pp. 235–240, 2015.

[2] H. Zuse, "Resolving the Mysteries of the Halstead Measures," in *Büren/Bundschuh/Dumke: Metrikon 2005 – Praxis der Software-Messung, Shaker-Verlag, Aachen, 2005*, 2005, pp. 107–122.

[3] J. Moreno-Leon, G. Robles, and M. Roman-Gonzalez, "Comparing computational thinking development assessment scores with software complexity metrics," *IEEE Glob. Eng. Educ. Conf. EDUCON*, vol. 10–13–Apri, no. April, pp. 1040–1045, 2016.

[4] M. J. P. Van Der Meulen and M. A. Revilla, "Correlations between internal software metrics and software dependability in a large population of small C/C++ programs," in *Proceedings - International Symposium on Software Reliability Engineering, ISSRE*, 2007, pp. 203–208.

[5] R. E. Al-Qutaish and A. Abran, "An Analysis of the Design and Definitions of Halstead ' s Metrics," *Proc. 15th Int. Work. Softw. Meas. IWSM2005*, pp. 337–352, 2005.

[6] D. Y. Pawade, M. Metha, K. Shah, and J. Rathod, "Python Based Software Complexity Calculator using Halstead Metrics," *Int. J. Adv. Found. Res. Comput.*, vol. 2, no. Special Issue (NCRTIT 2015), pp. 390–394, 2015.

[7] A. I. Cornwell, "Software Maintainability," *A Br. Overv. Appl. to LFEV 2015 Adam*, 2015.

[8] R. Sehgal and D. Mehrotra, "Predicting faults before testing phase using Halstead's metrics," *Int. J. Softw. Eng. its Appl.*, vol. 9, no. 7, pp. 135–142, 2015.

[9] T. Hariprasad, C. Thirumalai, G. Vidhyagaran, and K. Seenu, "Software Complexity Analysis Using Halstead Metrics," in *International Conference on Trends in Electronics and Informatics ICEI 2017*, 2017, no. May, pp. 1109–1112.

[10] C. Thirumalai, S. R.R., and R. R. L, "An Assessment of Halstead and COCOMO Model for Effort Estimation," in *International Conference on Innovations in Power and Advanced Computing Technologies [i-PACT2017]*, 2017, no. April, pp. 1–4.

[11] H. Ramasubramanian, "Halstead Metrics Tool: Software Metrics Analyzer for Java Programs," 2016. [Online]. Available: https://sourceforge.net/projects/halsteadmetricstool/files/CLI/v2.0/. [Accessed: 05-Jan-2018].

[12] P. Singh, K. D. Chaudhary, and S. Verma, "An Investigation of the Relationships between Software Metrics and Defects," *Int. J. Comput. Appl.*, vol. 28, no. 8, pp. 13–17, Aug. 2011.

[13] D. Spinellis, "Tool Writing: A Forgotten Art?," *IEEE Softw.*, vol. 22, no. 4, pp. 9–11, Jul. 2005.

[14] I. Binanto, H. L. H. S. Warnars, F. L. Gaol, E. Abdurachman, and B. Soewito, "Measuring the Quality of Various Version an Object-Oriented Software Utilizing CK Metrics," in *2018 International Conference on Information and Communications Technology (ICOIACT)*, 2018, pp. 41–44.

[15] F. Pérez, B. E. Granger, and J. D. Hunter, "Python: An ecosystem for scientific computing," *Comput. Sci. Eng.*, vol. 13, no. 2, pp. 13–21, 2011.

[16] S. Misra and F. Cafer, "Estimating complexity of programs in python language," *Tech. Gaz.*, vol. 18, pp. 23–32, 2011.

[17] M. Shudrak and V. V. Zolotarev, "Improving Fuzzing Using Software Complexity Metrics," in *Information Security and Cryptology - ICISC 2015: 18th International Conference, Seoul, South Korea, November 25-27, 2015, Revised Selected Papers*, 2016, no. March, pp. 246–261.

[18] I. Samoladas, I. Stamelos, L. Angelis, and A. Oikonomou, "Open Source Software Development Should Strive for EVEN GREATER CODE MAINTAINABILITY," *Commun. ACM*, vol. 47, no. 10, pp. 83–87, 2004.

[19] Python Software Foundation, "The Python Tutorial — Python 2.7.14 documentation." [Online]. Available: https://docs.python.org/2/tutorial/index.html. [Accessed: 09-Jan-2018].

[20] M. F. Sanner, "The Python interpreter as a framework for integrating scientific computing software-components," *Scripps Res. Inst.*, vol. 26, no. 1, pp. 1–12, 2008.

## halstead-undip

ORIGINALITY REPORT

| 14% | 5% | 12% | 4% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | Jesus Moreno-Leon, Gregorio Robles, Marcos Roman-Gonzalez. "Comparing computational thinking development assessment scores with software complexity metrics", 2016 IEEE Global Engineering Education Conference (EDUCON), 2016<br>Publication | 2% |
| 2 | T Hariprasad, G Vidhyagaran, K Seenu, Chandrasegar Thirumalai. "Software complexity analysis using halstead metrics", 2017 International Conference on Trends in Electronics and Informatics (ICEI), 2017<br>Publication | 1% |
| 3 | ceur-ws.org<br>Internet Source | 1% |
| 4 | zenodo.org<br>Internet Source | 1% |
| 5 | Iwan Binanto, Harco Leslie Hendric Spits Warnars, Ford Lumban Gaol, Edi Abdurachman, Benfano Soewito. "Measuring | 1% |

|  |  |  |
|---|---|---|
|  | the quality of various version an object-oriented software utilizing CK metrics", 2018 International Conference on Information and Communications Technology (ICOIACT), 2018<br>Publication |  |
| 6 | Yulius Denny Prabowo, Harco Leslie Hendric Spits Warnas, Ford Lumban Gaol, Edi Abdurachman, Benfano Soewito. "An initial research on Halstead's technique for pattern similarity relationship study", 2018 International Conference on Information and Communications Technology (ICOIACT), 2018<br>Publication | 1% |
| 7 | Submitted to Universiti Putra Malaysia<br>Student Paper | 1% |
| 8 | Rhio Sutoyo, Harco Leslie Hendric Spits Warnars, Ford Lumban Gaol, Edi Abdurachman, Benfano Soewito. "Measurement of QuestDone mobile application using 7 steps use case points method", 2017 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom), 2017<br>Publication | 1% |
| 9 | Urszula Markowska-Kaczmar. "The Influence of Using Design Patterns on the Process of Implementing Genetic Algorithms", Lecture | 1% |

| | | |
|---|---|---|
| | Notes in Computer Science, 2010<br>Publication | |
| 10 | indico.cern.ch<br>Internet Source | 1% |
| 11 | MORASCA, SANDRO. "SOFTWARE MEASUREMENT", Handbook of Software Engineering and Knowledge Engineering In 2 Volumes, 2001.<br>Publication | <1% |
| 12 | V.L. Loh. "Evaluation of query languages with software science metrics", Proceedings of TENCON 94 - 1994 IEEE Region 10 s 9th Annual International Conference on Frontiers of Computer Technology TENCON-94, 1994<br>Publication | <1% |
| 13 | Submitted to University of Philadelphia - Jordan<br>Student Paper | <1% |
| 14 | www.gelog.etsmtl.ca<br>Internet Source | <1% |
| 15 | www.software-improvers.nl<br>Internet Source | <1% |
| 16 | 139.78.48.197<br>Internet Source | <1% |
| 17 | C.L. Carnal. "An exploratory study on a linear model for measuring software quality", | <1% |

Proceedings IEEE Energy and Information Technologies in the Southeast, 1989
Publication

| | | | |
|---|---|---|---|
| Exclude quotes | On | Exclude matches | Off |
| Exclude bibliography | On | | |

# halstead-undip

PAGE 1

**Article Error** You may need to use an article before this word. Consider using the article **the**

**Article Error** You may need to use an article before this word. Consider using the article **the**

**Article Error** You may need to use an article before this word. Consider using the article **the**

**Article Error** You may need to use an article before this word. Consider using the article **the**

**Wrong Article** You may have used the wrong article or pronoun. Proofread the sentence to make sure that the article or pronoun agrees with the word it describes.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**S/V** This subject and verb may not agree. Proofread the sentence to make sure the subject agr with the verb.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Confused** You have used **an** in this sentence. You may need to use **a** instead.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Dup.** You have typed two **articles** in a row. You may need to delete one of them.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**S/V** This subject and verb may not agree. Proofread the sentence to make sure the subject agr with the verb.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Article Error** You may need to use an article before this word.

**Article Error** You may need to remove this article.

**Missing ","** You have a spelling or typing mistake that makes the sentence appear to have a comma error.

**Article Error** You may need to use an article before this word.

**S/V** This subject and verb may not agree. Proofread the sentence to make sure the subject agr with the verb.

**Wrong Article** You may have used the wrong article or pronoun. Proofread the sentence to make sure that the article or pronoun agrees with the word it describes.

**P/V** You have used the passive voice in this sentence. Depending upon what you wish to emphasize in the sentence, you may want to revise it using the active voice.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Proofread** This part of the sentence contains a grammatical error or misspelled word that makes your meaning unclear.

**P/V** You have used the passive voice in this sentence. Depending upon what you wish to emphasize in the sentence, you may want to revise it using the active voice.

**Article Error** You may need to use an article before this word.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Confused** You have used **a** in this sentence. You may need to use **an** instead.

PAGE 2

**Article Error** You may need to use an article before this word.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Article Error** You may need to use an article before this word.

**Article Error** You may need to use an article before this word. Consider using the article **the**

**Wrong Article** You may have used the wrong article or pronoun. Proofread the sentence to make sure that the article or pronoun agrees with the word it describes.

**Article Error** You may need to use an article before this word.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Proofread** This part of the sentence contains a grammatical error or misspelled word that ma your meaning unclear.

**P/V** You have used the passive voice in this sentence. Depending upon what you wish to emphasize in the sentence, you may want to revise it using the active voice.

**Article Error** You may need to use an article before this word.

**Article Error** You may need to use an article before this word.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Article Error** You may need to use an article before this word.

**Article Error** You may need to use an article before this word.

PAGE 3

**Article Error** You may need to use an article before this word.

**Article Error** You may need to use an article before this word.

**Prep.** You may be using the wrong preposition.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Article Error** You may need to use an article before this word.

**S/V** This subject and verb may not agree. Proofread the sentence to make sure the subject agr with the verb.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Article Error** You may need to use an article before this word.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Article Error** You may need to use an article before this word. Consider using the article **a**.

**Wrong Article** You may have used the wrong article or pronoun. Proofread the sentence to make sure that the article or pronoun agrees with the word it describes.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Article Error** You may need to use an article before this word. Consider using the article **the**

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Article Error** You may need to use an article before this word.

PAGE 4

---

**Missing ","** You may need to place a comma after this word.

**Sentence Cap.** Remember to capitalize the first word of each sentence.

**Verb** This verb may be incorrect. Proofread the sentence to make sure you have used the corr form of the verb.

**Missing ","** You may need to place a comma after this word.

**Possessive** This word may be a plural noun and may not need an apostrophe.

**Frag.** This sentence may be a fragment or may have incorrect punctuation. Proofread the sent to be sure that it has correct punctuation and that it has an independent clause with a complete subject and predicate.