**De La Salle University - Dasmariñas**

**DBB-B Dasmariñas, Cavite, Philippines**

**West Ave, Dasmariñas, Cavite**

Jayag, Reinon Vien Irvinz
Morales, Lincoln
Reyes, Jerizsa Hannah
BCS21

**Research Title: Code Readability Comparison between Java and Python**

**Research Description:**

Code readability is a very important factor when making competent software or programs. It determines the ease of knowing why the software is not working properly or how to better improve upon the software. Some programmers do not bother making their code cleaner and easier to understand as long as the program created fulfills its predetermined function. With this in mind, the researchers created this research to determine which programming language among Java and Python is more readable to programmers. The basis for readability will mainly be in the field of semantics and how lines of code should behave in each language. Comments on source code will be used to correlate meaning.

The paper will use Dorn's model, Scalabrino's model, and Dhabhai's model for code readability. In Dorn's model, code readability is defined by four defining categories which are: Visual features, Spatial features, Alignment features, and Natural- language features (Scalabrino, S., Linares-Vásquez, M., Oliveto, R., & Poshyvanyk, D., 2018). The model used in Scalabrino research on the other hand follows a procedure in order to determine the readability of a source code. The procedure involves removing non-textual tokens from the corpora operators, splitting the remaining tokens into separate words by using the underscore or camelcase separators, removing words that belong to a stop-word list and extracting stems from words by using the Porter algorithm. (Scalabrino, S., Linares-Vásquez, M., Oliveto, R., & Poshyvanyk, D., 2018). Lastly, Dhabai's model relies on seven rules for calculating readability. These rules are: total lines of code, line length, number of comment line, number of blank lines, number of lines after semicolon, number of space after directive statements and number of methods. (Dhabhai, D., Dua, A.K., Saroliya, A., Purohit, R., 2015)

## Algorithm Rules

### Code Comparison Algorithm

1. Gather source code data of two languages, Python and Java
2. Calculate the Python and Java's source code's readability results using Random Forest Algorithm.
3. Examine and compare results using Dorn, Scalabrino, and Dhabai's models.
4. Interpret and analyze final result and repeat for other programs of Python and Java.

Scalabrino's model for code readability

1.) Comments and Identifiers Consistency (CIC)

$$CIC(m) = \frac{|Comments(m) \cap Ids(m)|}{|Comments(m) \cup Ids(m)|}$$

Comments = Comments
Ids = Identifiers
m = methods

2.) Identifier Terms in Dictionary (ITID)

$$ITID(l) = \frac{|Terms(l) \cap Dictionary|}{|Terms(l)|}$$

Terms = Terms
Dictionary = Words in English Language
l = line of a method

3.) Narrow Meaning Identifiers (NMI)

$$NMI(l) = \sum_{t \in l} particularity(t)$$

particularity = Terms
t = number of hops t to the root node in the hypernym tree of t
l = line of code

Flesch–Kincaid readability test

$$206.835 - 1.015\left(\frac{\text{total words}}{\text{total sentences}}\right) - 84.6\left(\frac{\text{total syllables}}{\text{total words}}\right)$$

Dhabhai's model for code readability
According to Dhabai (Dhabhai, D., Dua, A.K., Saroliya, A., Purohit, R., 2015), in order to determine code readability, the total lines of code should be added together with: line length, number of comment line, number of blank line, number of lines after semicolon, number of space after directive statements and number of methods used.

**Complexity of Algorithm**

The complexity of the Random Forest Algorithm is based on the number of attributes and training samples that will be used. Since we have not yet tested the source code, we won't be entirely sure what the time and space complexity of the algorithm will be. However, we have found that the complexity of building a tree with m as attributes and n as the number of trees is O( m*n log n).

**References**

Börstler, J., & Paech, B. (2016). The role of method chains and comments in software readability and comprehension—an experiment. IEEE Transactions on Software Engineering, 42(9), 886-898. DOI: 10.1109/TSE.2016.2527791

Chen, H., Huang, Y., Liu, Z., Chen, X., Zhou, F., & Luo, X. (2019). Automatically detecting the scopes of source code comments. Journal of Systems and Software, 153, 45-63. DOI: https://doi.org/10.1016/j.jss.2019.03.010

Dhabhai, D., Dua, A.K., Saroliya, Anil, & Purohit, R. (2015). Evaluation of Quality of Source Code By Code Readability. Retrieved from http://ijarcsse.com/Before_August_2017/docs/papers/Volume_5/10_October2015/V5I10-0110.pdf. ISSN: 2277 128X

dos Santos, R. M., & Gerosa, M. A. (2018, May). Impacts of coding practices on readability. In Proceedings of the 26th Conference on Program Comprehension (pp. 277-285). DOI: https://doi.org/10.1145/3196321.3196342

Donges, N. (2019, June 16). A complete guide to the random forest algorithm. Retrieved March 1, 2020, from https://builtin.com/data-science/random-forest-algorithm#real

Fakhoury, S., Roy, D., Hassan, A., & Arnaoudova, V. (2019, May). Improving source code readability: theory and practice. In 2019 IEEE/ACM 27th International Conference on Program Comprehension (ICPC) (pp. 2-12). IEEE. DOI: 10.1109/ICPC.2019.00014

Johnson, J., Lubo, S., Yedla, N., Aponte, J., & Sharif, B. An Empirical Study Assessing Source Code Readability in Comprehension. In 2019 IEEE International Conference on Software Maintenance and Evolution (ICSME) (pp. 513-523). IEEE. DOI: 10.1109/ICSME.2019.00085

Scalabrino, S., Linares-Vásquez, M., Oliveto, R., & Poshyvanyk, D. (2018). A comprehensive model for code readability. Journal of Software: Evolution and Process, 30(6). doi: 10.1002/smr.1958

Sedano, T. (2016, April). Code readability testing, an empirical study. In 2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET) (pp. 111-117). IEEE. DOI: 10.1109/CSEET.2016.36

Song, X., Sun, H., Wang, X., & Yan, J. (2019). A survey of automatic generation of source code comments: Algorithms and techniques. IEEE Access, 7, 111411-111428. DOI: 10.1109/ACCESS.2019.2931579