Technologica

Vol.4(2): Hal 170-180 (Juli 2025)

ISSN: 2827-9492 (*Online*)

Analisis Dampak *Blank Lines Of Code* Terhadap Efisiensi Memori dan Energi

Impact Analysis of Blank Lines Of Code on Memory and Energy Efficiency

Inayatullah[*,1], Nyimas Nisrinaa Kamilah[2]

[1]Program Studi Manajemen Informatika, Fakultas Ilmu Komputer dan Rekayasa, Universitas Multi Data Palembang, Palembang, Indonesia
[2]Program Studi Informatika, Fakultas Ilmu Komputer dan Rekayasa, Universitas Multi Data Palembang, Palembang, Indonesia

[*]Inayatullah
Email: inayatullah@mdp.ac.id

Abstrak. Pertumbuhan transformasi digital yang sangat pesat dalam dekade terakhir menuntut efisiensi tidak hanya dari sisi fungsionalitas perangkat lunak, tetapi juga dari penggunaan sumber daya seperti memori dan energi. Salah satu elemen dalam kode program yang sering diabaikan namun berpotensi mempengaruhi efisiensi adalah baris kosong atau *blank lines of code* (BLoC). Pendekatan *green software* menjadi salah satu solusi yang banyak dikembangkan untuk menjawab tantangan ini. Penelitian ini secara khusus bertujuan untuk membahas dampak BLoC terhadap efisiensi penggunaan memori dan energi. Pendekatan yang digunakan adalah pendekatan *clean code*, yang meliputi tahapan pemahaman kode, penghitungan nilai kode sebelum dan sesudah penghapusan BLoC, serta analisis hasil pengujian. Hasil penelitian menunjukkan adanya efisiensi pada penggunaan memori dan energi setelah BLoC dihapus dari kode program. Sebelum dilakukan pembersihan (*clean*), penggunaan memori tercatat sebesar 5060,4 KB dan konsumsi energi sebesar 0,0023 kWh. Setelah BLoC dihapus, penggunaan memori menurun menjadi 5011,28 KB dan konsumsi energi menjadi 0,0022 kWh. Berdasarkan hasil pengujian dapat disimpulkan bahwa penggunaan *blank lines of code* dalam program dapat mempengaruhi efisiensi sumber daya. Penghapusan BLoC terbukti dapat membantu mengoptimalkan performa perangkat lunak dari sisi penggunaan memori dan energi.
Kata kunci: Universitas Multi Data Palembang, Efisiensi, Memori, Energi, *Clean Code*

Abstract. *The rapid growth of digital transformation over the past decade demands efficiency not only in terms of software functionality but also in the use of resources such as memory and energy. One element in program code that is often overlooked but has the potential to affect efficiency is blank lines of code (BLoC). The green software approach has emerged as a widely developed solution to address this challenge. This study specifically aims to examine the impact of BLoC on memory and energy efficiency. The approach used is the clean code method, which includes stages such as code comprehension, measuring code values before and after BLoC removal, and analyzing the test results. The findings show that there is efficiency in memory and energy usage after BLoC is removed from the program code. Before the clean process, memory usage was recorded at 5060.4 KB and energy consumption at 0.0023 kWh. After BLoC was removed, memory usage decreased to 5011.28 KB and energy consumption to 0.0022 kWh. Based on the test results, it can be concluded that the use of blank lines of code in a program can affect resource efficiency. The removal of BLoC has been proven to help optimize software performance in terms of memory and energy usage.*
Keywords: *Blank Lines of Code, Efficiency, Memory, Energy, Clean Code*

1. Pendahuluan

Pertumbuhan transformasi digital yang sangat pesat dalam dekade terakhir menuntut adanya efisiensi tidak hanya dari sisi fungsionalitas perangkat lunak, tetapi juga dari segi penggunaan sumber daya seperti memori dan energi. Pertumbuhan transformasi digital telah memberikan manfaat yang signifikan, tetapi sektor energi relatif lambat (Maroufkhani et al., 2022)(Nazaré et al., 2023). Pemikiran *green software* menjadi salah satu pendekatan yang banyak dikembangkan untuk menjawab tantangan ini. *Green software* menekankan pada penerapan-penerapan pengembangan perangkat lunak yang ramah lingkungan, termasuk dalam hal efisiensi memori dan energi selama proses komputasi (Hilty & Aebischer, 2015)(Verdecchia et al., 2021).

Salah satu elemen dalam program yang sering kali kurang dipertimbangkan, namun berpotensi mempengaruhi efisiensi adalah *Blank Lines of Code* (BLoC) (Sarhan, 2019). BLoC digunakan untuk meningkatkan keterbacaan dan organisasi visual dalam penulisan kode, terutama dalam tim pengembang yang besar (Ramasubbu & Kemerer, 2016). Meskipun tidak berisi instruksi eksekusi, BLoC tetap diproses oleh *compiler* atau *interpreter*, dan dapat mempengaruhi waktu proses serta konsumsi sumber daya seperti memori dan energi pada saat eksekusi (Georgiou et al., 2018).

Beberapa penelitian menyatakan bahwa efisiensi perangkat lunak tidak hanya dipengaruhi oleh kompleksitas algoritma, namun juga oleh bagaimana kode disusun dan dikelola (Procaccianti et al., 2016) seperti kode dengan struktur yang panjang dan tidak optimal dapat memperbesar ukuran *file*, kurangnya optimasi kueri di database dapat memperlambat waktu load, serta berdampak pada penggunaan *Central Processing Unit* (CPU) dan memori. Dalam skenario di mana perangkat lunak dijalankan berulang kali atau dalam jangka panjang, hal-hal kecil seperti penggunaan BLoC yang tidak perlu dapat menambah memori dan konsumsi energi secara akumulatif. Oleh karena itu perangkat lunak membutuhkan pemeliharaan yang baik (Mancebo et al., 2021).

Penelitian yang secara khusus bertujuan membahas dampak *Blank Lines of Code* (BLoC) terhadap efisiensi memori dan energi masih sangat terbatas. Oleh karena itu, diperlukan penelitian yang fokus pada pengaruh elemen-elemen dalam kode seperti BLoC, terhadap performa perangkat lunak secara keseluruhan. Pendekatan eksperimental dalam pengujian penggunaan memori dan konsumsi energi berdasarkan variasi struktur kode menjadi penting untuk dilakukan(Ljung & Gonzalez-Huerta, 2022).

Melalui penelitian ini, diharapkan dapat ditemukan bukti berdasarkan percobaan, mengenai peran BLoC dalam efisiensi memori dan energi perangkat lunak, sehingga pengembang dapat mengambil keputusan yang tepat dalam menyeimbangkan antara keterbacaan dan performa kode. Dengan demikian, penerapan pengkodean yang sadar lingkungan dan keberlanjutan dapat lebih terintegrasi dalam proses pengembangan perangkat lunak masa kini (Raisian et al., 2018)(Pritchett, 2024).

2. Pendekatan

Pendekatan yang digunakan dalam penelitian ini menggunakan pendekatan *clean code*. Penelitian ini, memiliki tahapan identifikasi masalah dan tujuan penelitian, desain eksperimen, pengumpulan data, analisis data, interpretasi dan evaluasi hasil, serta kesimpulan dan rekomendasi. *Clean code* suatu pendekatan dimana kode yang dibangun harus mudah untuk dibaca, dimengerti, diubah, dan diuji oleh orang lain, termasuk programmer lain yang tidak

menulisnya. Menulis dengan pendekatan *clean code* dapat memberikan manfaat berupa peningkatan produktivitas, kualitas perangkat lunak, keterbacaan, dan keamanan produk (Digkas et al., 2022)(Filazzola & Lortie, 2022).

2.1 Jenis Penelitian

Penelitian ini merupakan penelitian kuantitatif eksperimental dengan pendekatan komparatif. Tujuannya adalah untuk mengukur dan membandingkan penggunaan memori dan efisiensi energi dari kode program yang mengandung *Blank Lines of Code* (BLoC) dalam jumlah tertentu dengan kode program yang telah dihapus *Blank Lines of Code* (BLoC). Penelitian ini difokuskan pada pengamatan *Blank Lines of Code* (BLoC) terhadap efisiensi sumber daya memori dan energi dari perangkat lunak.

2.2 Objek Penelitian

Objek dalam penelitian ini menggunakan dataset Tanjak yang berisi berkas kode program dalam bahasa pemrograman tertentu (VB, PHP, Java, C# dan database (SQL dan MySQL)). *File* yang digunakan berjumlah sebanyak 15 *file*.

2.3 Teknik Pengumpulan Data

Dalam proses pengumpulan data dilakukan dengan langkah-langkah sebagai berikut:

1. Mengumpulkan semua kode program dan database.
2. Mengubah semua kode program dan database dalam bentuk *file* txt.
3. Menggabung kode program dan database yang sudah ada dalam bentuk *file* txt.

2.4 Teknik Analisis Data

Data yang diperoleh dianalisis secara kuantitatif dengan menggunakan teknik *clean code*. Data yang diamati meliputi

File, *file* yang digunakan dalam penelitian ini sebanyak 15 *file* program yang berbeda.

Lines of Code = Banyaknya baris yang ada dalam *file* program yang digunakan, dihitung menggunakan aplikasi yang dibangun oleh author.

Memori (KB) = Jumlah memori yang digunakan oleh setiap *file* di dapatkan berdasarkan jumlah karakter (Byte) yang terdapat dalam *file*. Dihitung menggunakan aplikasi yang dibangun oleh author

Energi (J) = Byte * 8 bit * 0.0002 J dihitung menggunakan aplikasi yang dibangun author

Energi (kWh) = Energi (J)/3.600.000 dihitung menggunakan aplikasi yang dibangun author

Selanjutnya, dilakukan pengujian untuk melihat apakah terdapat perbedaan signifikan antara kode program dengan BLoC dan kode program tanpa BLoC terhadap memori dan konsumsi energi.

2.5 Alat dan Bahan

Alat dan bahan yang digunakan dalam penelitian meliputi:

a. Sistem operasi: Windows 10.
b. Perangkat keras: Laptop dengan prosesor Intel i7, RAM 16 GB.
c. Aplikasi untuk pengujian

2.6 Teknik Pengujian

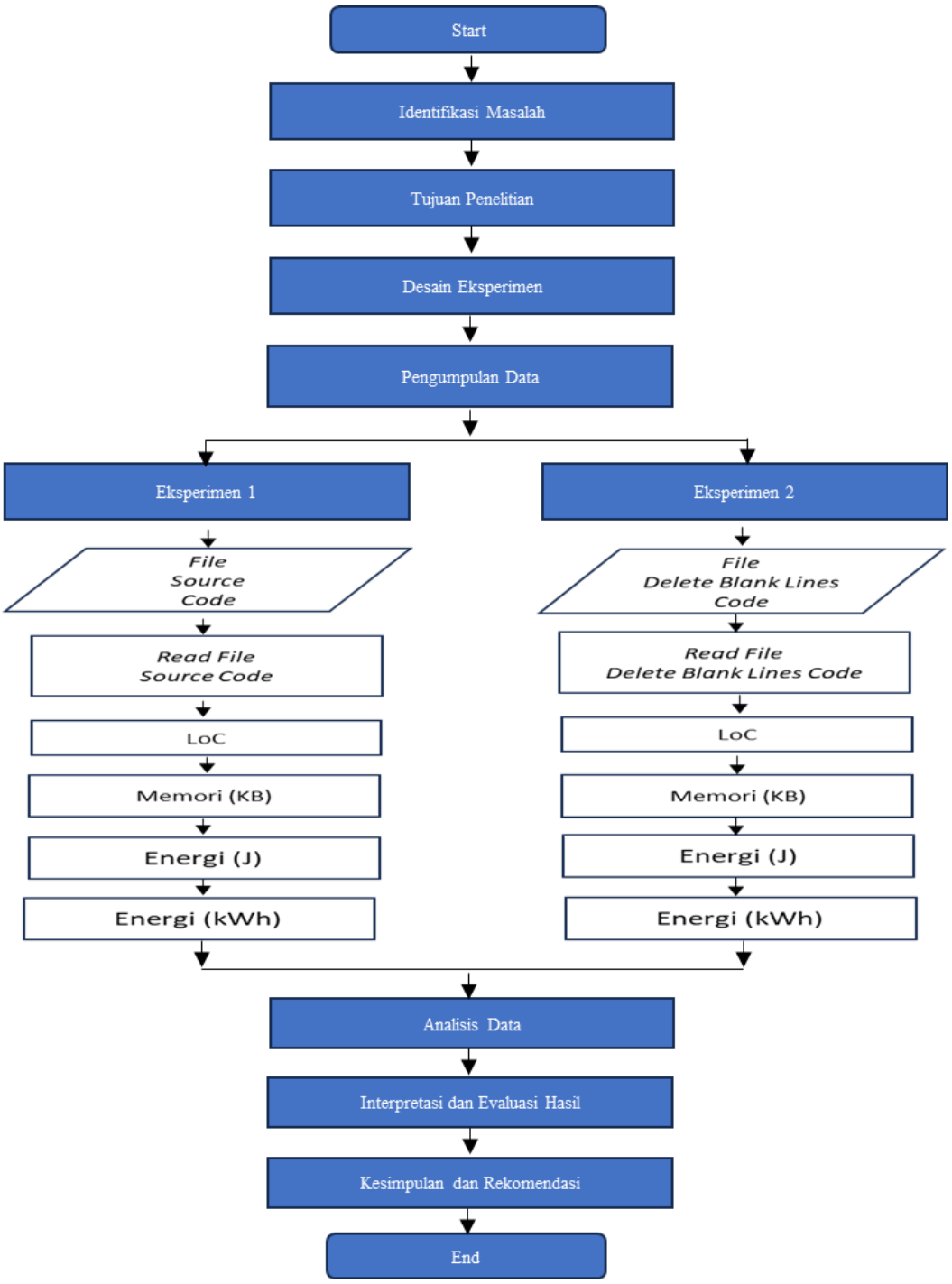
Teknik pengujian yang digunakan dalam peneliti dengan menggunakan langkah-langkah berikut:

1. Pengujian dilakukan dengan 15 kode program dan database yang berbeda.

2. Menghitung kode program dan database dengan *blank lines of code* (BLoC)
3. Menyimpan:
   a. Hasil Konsumsi energi (dalam satuan *Joule* dan kWh).
   b. Hasil Penggunaan memori (dalam satuan Kilo Byte (KB)).
4. Menghapus *blank lines of code* dari kode program dan database
5. Menghitung kode program dan database tanpa *blank line of code*
6. Menyimpan:
   a. Hasil Konsumsi energi (dalam satuan Joule atau kWh).
   b. Hasil Penggunaan memori (dalam satuan Kilo Byte (KB)).
7. Membandingkan hasil perhitungan kode program dan database dengan *blank lines of code* dengan kode program dan database tanpa *blank lines of code.*

2.7 Diagram Alir Penelitian

Diagram *flowchart* ini menggambarkan alir penelitian yang dilakukan. Selain berfungsi sebagai alat komunikasi, *flowchart* juga dapat digunakan sebagai panduan dalam pengembangan program (Khesya N., 2021). Diagram alir penelitian dapat dilihat pada Gambar 1.

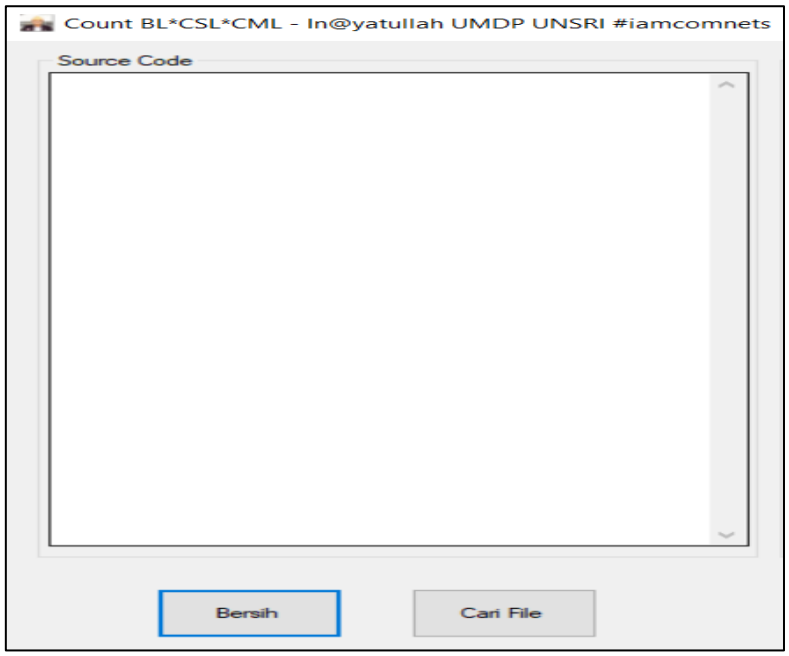


Gambar 1. Diagram Alir Penelitian

3. Hasil dan Pembahasan

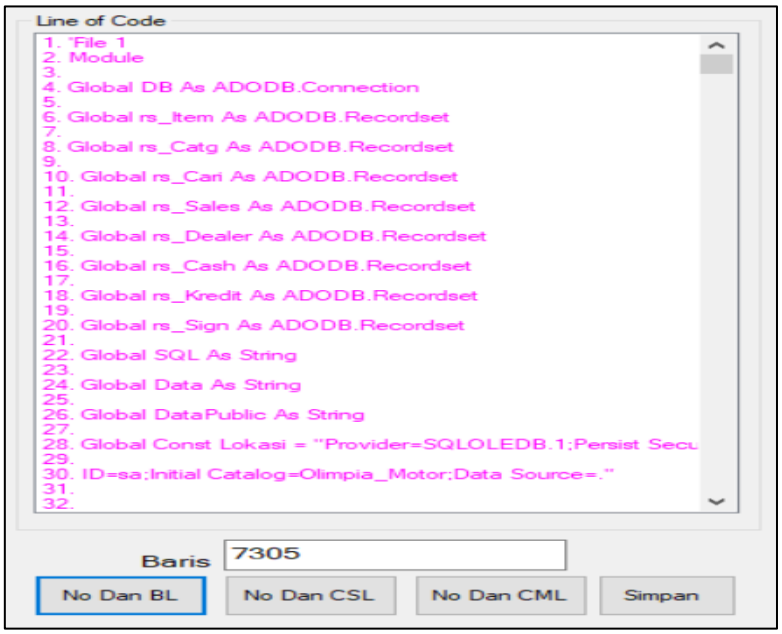
3.1 Hasil

Berdasarkan data yang ada dilakukan pengujian untuk mendapat hasil, yang akan digunakan untuk mengetahui dampak *Blank Line of Code* (BLoC) terhadap memori dan konsumsi energi yang digunakan. Pada Gambar 2, tampilan form yang digunakan untuk menampilkan kode program yang akan digunakan dalam pengujian.



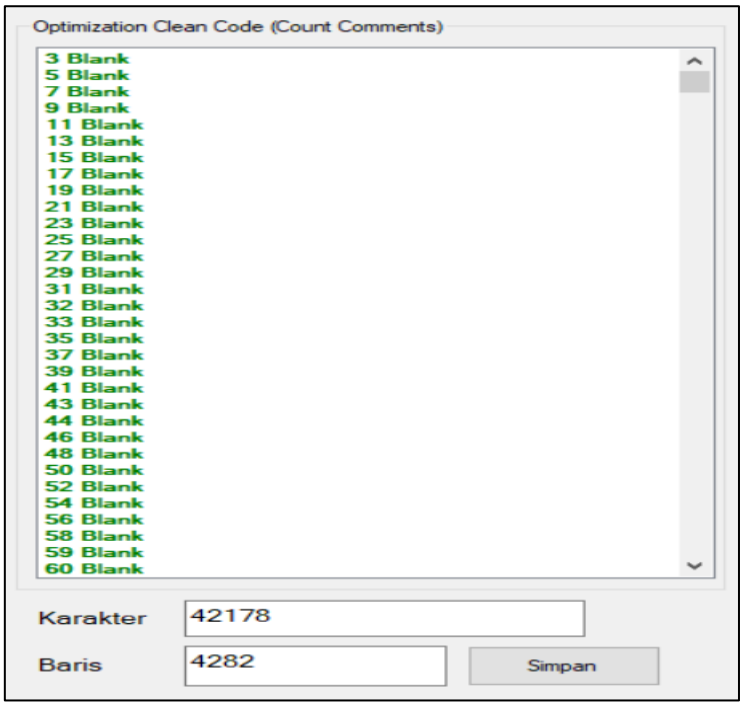
Gambar 2. Form Untuk Tampil Kode Program

Pada Gambar 3, tampilan form yang sudah berisi baris kode program berdasarkan pencarian *file* yang dilakukan oleh pengguna dan jumlah baris dari kode program. Dalam pengujian pertama, digunakan *file* 1.



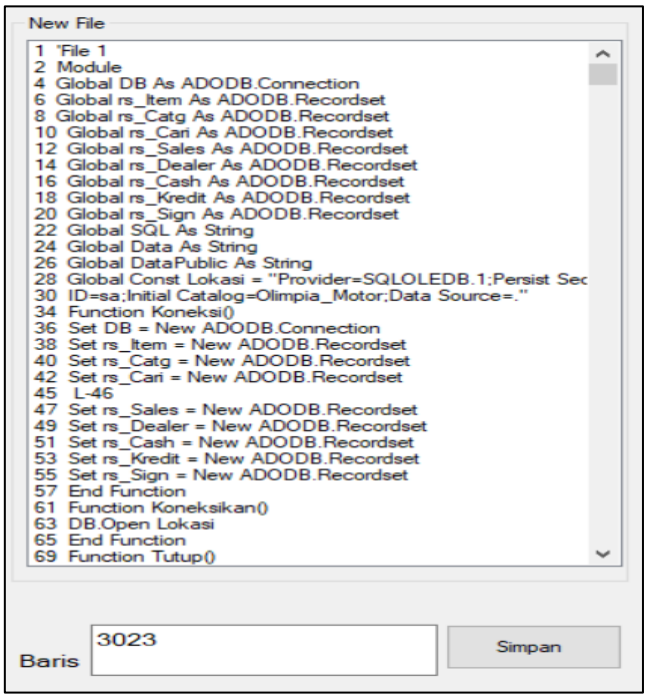
Gambar 3. Form Baris Kode Program

Pada gambar 4, tampilan form yang berisi *blank lines of code* dari kode program *file* 1 yang digunakan.dalam pengujian. Form berisi no baris dari *blank lines of code*, karakter, dan banyaknya jumlah baris dari BLoC.



Gambar 4. Form Tampilan *Blank Lines Of Code*

Pada gambar 5, tampilan form yang berisi kode program *file* 1 yang baru, tanpa *blank lines of code* dan jumlah baris yang baru.



Gambar 5. Form Tampilan *File* 1 Baru.

Selanjutnya dilakukan proses perhitungan terhadap 15 *file* yang digunakan dalam pengujian, dengan data-data yang telah ditentukan (*File*, LoC, Memori, Energi (J), dan Energi (kWh). Pada Tabel 1, berisi hasil perhitungan dari 15 *file* kode program yang digunakan tanpa menghilangkan BLoC dan pada Tabel 2 berisi hasil pengujian dengan menghilangkan BLoC.

Tabel 1. Hasil Pengujian terhadap *File-File* Tanpa Menghilangkan BLoC

Source Code				
File	LoC	Memori (KB)	Energi (J)	Energi (kWh)
1-P.txt	7305	99,17	162,4864	0,0000451351
2-P.txt	8361	402,61	659,6368	0,0001832324
3-P.txt	2306	60,41	98,9824	0,0000274951
4-P.txt	3854	156,12	255,7248	0,0000710347
5-P.txt	1812	56,73	92,9504	0,0000258196
6-P.txt	11203	455,18	745,5056	0,0002070849
7-P.txt	4063	125,75	206,0208	0,0000572280
8-P.txt	29425	1513,3	2479,1376	0,0006886493
9-P.txt	3733	91,79	150,3936	0,0000417760
10-P.txt	13748	578,72	947,9232	0,0002633120
11-P.txt	3415	106,72	174,8528	0,0000485702
12-P.txt	8686	438,14	717,7712	0,0001993809
13-P.txt	17634	711,27	1165,1024	0,0003236396
14-P.txt	4216	170,96	280,0528	0,0000777924
15-P.txt	3130	93,53	153,2432	0,0000425676
Total	122891	5060,4	8289,784	0,002302718

Tabel 2. Hasil Pengujian terhadap *File*-File Dengan Menghilangkan BLoC

Source Code				
File	Hapus BLoC	Memory BLoC KB)	Energy (J) BLoC	Energy (kWh) BLoC
1-P.txt	3023	88,58	145,1328	0,0000403147
2-P.txt	7240	400,32	655,8896	0,0001821916
3-P.txt	2085	59,98	98,2752	0,0000272987
4-P.txt	3632	155,57	254,8128	0,0000707813
5-P.txt	1682	56,48	92,5344	0,0000257040
6-P.txt	10224	452,2	740,6192	0,0002057276
7-P.txt	4056	125,73	205,9984	0,0000572218
8-P.txt	27045	1496,17	2451,0672	0,0006808520
9-P.txt	3405	91,15	149,3408	0,0000414836
10-P.txt	12667	575,53	942,6912	0,0002618587
11-P.txt	3413	106,72	174,8496	0,0000485693
12-P.txt	7988	432,69	708,8464	0,0001969018
13-P.txt	16418	708,7	1160,8896	0,0003224693
14-P.txt	3788	169,84	278,2144	0,0000772818
15-P.txt	2267	91,62	150,1104	0,0000416973
Total	108933	5011,28	8209,272	0,0022803533

.

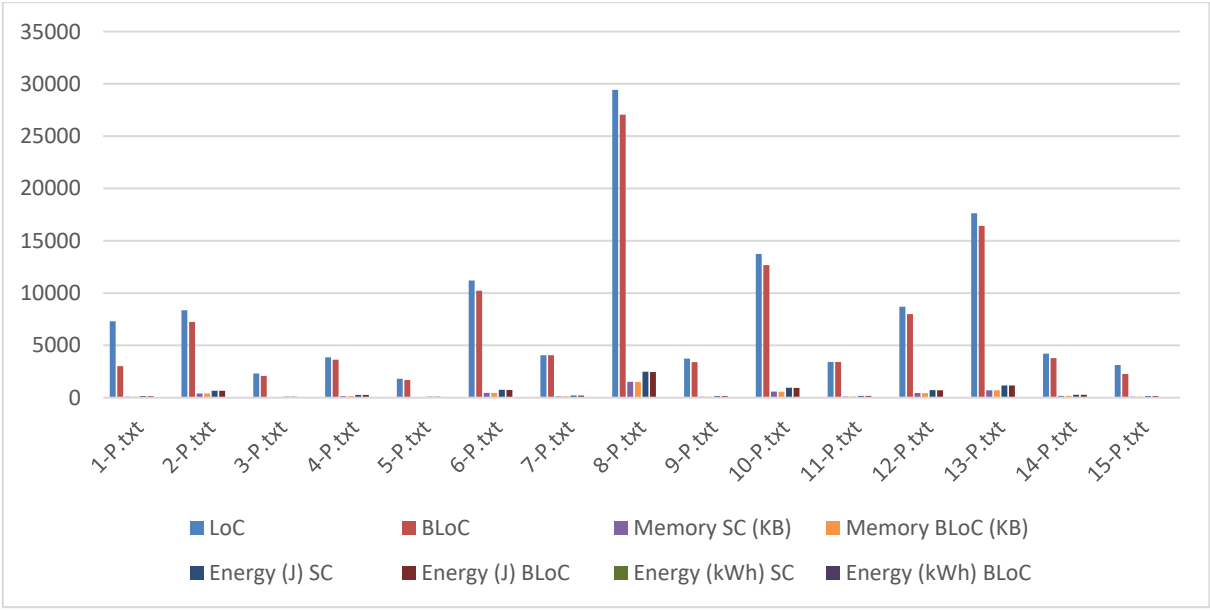
3.2 Pembahasan

Berdasarkan hasil dari pengujian yang telah dilakukan terhadap 15 *file*, dapat dilihat pada Tabel 3 dan grafik 6 terdapat perbandingan terhadap LoC, memori yang digunakan, konsumsi energi (J) dan konsumsi energi (kWh). Contoh Pada Tabel 3, file 1 tanpa menghilangkan BLoC dihasilkan LoC sebesar 7305, memori sebesar 99,17 KB, energi (J) sebesar 162,4864, dan energi (kWh) sebesar 0,0000451351. Setelah dilakukan pengujian dengan menghilangkan BLoC

dihasilkan LoC sebesar 3023, memori sebesar 88,58 KB, energi (J) sebesar 145,1328, dan energi (kWh) sebesar 0,0000403147.

Tabel 3. Perbandingan Data LoC, Memori, Energi (J), dan Energi (kWh)

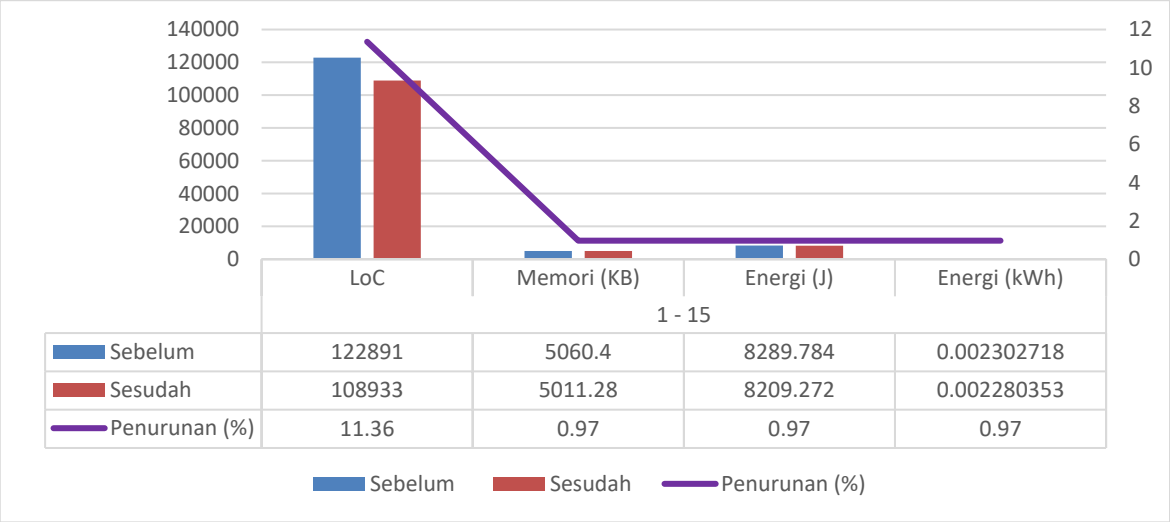
Source Code								
File	LoC	Hapus BLoC	Memory SC (KB)	Memory BLoC (KB)	Energi (J) SC	Energi (J) BLoC	Energi (kWh) SC	Energi (kWh) BLoC
1-P.txt	7305	3023	99,17	88,58	162,4864	145,1328	0,0000451351	0,0000403147
2-P.txt	8361	7240	402,61	400,32	659,6368	655,8896	0,0001832324	0,0001821916
3-P.txt	2306	2085	60,41	59,98	98,9824	98,2752	0,0000274951	0,0000272987
4-P.txt	3854	3632	156,12	155,57	255,7248	254,8128	0,0000710347	0,0000707813
5-P.txt	1812	1682	56,73	56,48	92,9504	92,5344	0,0000258196	0,0000257040
6-P.txt	11203	10224	455,18	452,2	745,5056	740,6192	0,0002070849	0,0002057276
7-P.txt	4063	4056	125,75	125,73	206,0208	205,9984	0,0000572280	0,0000572218
8-P.txt	29425	27045	1513,30	1496,17	2479,1376	2451,0672	0,0006886493	0,0006808520
9-P.txt	3733	3405	91,79	91,15	150,3936	149,3408	0,0000417760	0,0000414836
10-P.txt	13748	12667	578,72	575,53	947,9232	942,6912	0,0002633120	0,0002618587
11-P.txt	3415	3413	106,72	106,72	174,8528	174,8496	0,0000485702	0,0000485693
12-P.txt	8686	7988	438,14	432,69	717,7712	708,8464	0,0001993809	0,0001969018
13-P.txt	17634	16418	711,27	708,7	1165,1024	1160,8896	0,0003236396	0,0003224693
14-P.txt	4216	3788	170,96	169,84	280,0528	278,2144	0,0000777924	0,0000772818
15-P.txt	3130	2267	93,53	91,62	153,2432	150,1104	0,0000425676	0,0000416973
Total	122891	108933	5060,40	5011,28	8289,784	8209,272	0,0023027178	0,0022803533



Gambar 6. Perbandingan Kode Program dengan BLoC dan Tanpa BLoC

Tabel 4. Hasil Pengujian Sebelum dan Sesudah Penerapan BLoC

File	*Source Code*	Sebelum	Sesudah	Penurunan (%)
1-15	BLoC	122891	108933	11,36
	Memori (KB)	5060,4	5011,28	0,97
	Energi (J)	8289,784	8209,272	0,97
	Energi (kWh)	0,002302718	0,002280353	0,97



Gambar 7. Grafik Hasil Pengujian Sebelum dan Sesudah BLoC

4. Kesimpulan

Penggunaan *blank lines of code* pada kode program dapat mempengaruhi pemakaian sumber daya seperti LoC, memori dan energi. Pada Tabel 4 dan grafik 7, berdasarkan hasil pengujian terjadi penurunan penggunaan LoC, memori, energi (J), dan energi (kWh). Penurunan untuk LoC 11,36, memori 0,97, energi (J) 0,97, energi (kWh) 0,97. Untuk itu, dalam melakukan pembuatan kode program perlu memperhatikan barisan kode program *blank lines of code* guna menciptakan kode program yang hemat memori, konsumsi energi, dan berkelanjutan.

Daftar Pustaka

Digkas, G., Chatzigeorgiou, A., Ampatzoglou, A., & Avgeriou, P. (2022). Can Clean New Code Reduce Technical Debt Density? *IEEE Transactions on Software Engineering*, *48*(5), 1705–1721. https://doi.org/10.1109/TSE.2020.3032557

Filazzola, A., & Lortie, C. J. (2022). A call for clean code to effectively communicate science. *Methods in Ecology and Evolution*, *13*(10), 2119–2128. https://doi.org/10.1111/2041-210X.13961

Georgiou, K., Blackmore, C., Xavier-de-Souza, S., & Eder, K. (2018). Less is more: Exploiting the standard compiler optimization levels for better performance and energy consumption. *Proceedings of the 21st International Workshop on Software and Compilers for Embedded Systems, SCOPES 2018*, 35–42. https://doi.org/10.1145/3207719.3207727

Hilty, L., & Aebischer, B. (2015). ICT Innovations for Sustainability. *Advances in Intelligent*

Systems and Computing, *310*, 351–365. https://doi.org/10.1007/978-3-319-09228-7

Khesya N. (2021). *Mengenal Flowchart Dan Pseudocode*. http://doi.org/10.31219/osf.io/dq45c

Ljung, K., & Gonzalez-Huerta, J. (2022). "To Clean Code or Not to Clean Code" A Survey Among Practitioners. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *13709 LNCS*, 298–315. https://doi.org/10.1007/978-3-031-21388-5_21

Mancebo, J., Calero, C., & García, F. (2021). Does maintainability relate to the energy consumption of software? A case study. *Software Quality Journal*, *29*(1), 101–127. https://doi.org/10.1007/s11219-020-09536-9

Maroufkhani, P., Desouza, K. C., Perrons, R. K., & Iranmanesh, M. (2022). Digital transformation in the resource and energy sectors: A systematic review. *Resources Policy*, *76*. https://doi.org/10.1016/j.resourpol.2022.102622

Nazaré, T., Gadelha, J., Nepomuceno, E., & Lozi, R. (2023). Green Computing for Energy Transition: A Survey. *IEEE Latin America Transactions*, *21*(9), 937–948. https://doi.org/10.1109/tla.2023.10251799

Pritchett, S. (2024). *Energy Efficiency and Sustainability : The Intersection of Green Software Energy Efficiency and Sustainability : The Intersection of Green Software Engineering and Renewable Energy Date : October , 2024. October*. https://doi.org/10.13140/RG.2.2.35566.22089

Procaccianti, G., Fernández, H., & Lago, P. (2016). Empirical evaluation of two best practices for energy-efficient software development. *Journal of Systems and Software*, *117*, 185–198. https://doi.org/10.1016/j.jss.2016.02.035

Raisian, K., Yahaya, J., & Deraman, A. (2018). Exploring potential factors in green and sustainable software product. *International Journal on Informatics Visualization*, *2*(1), 23–27. https://doi.org/10.30630/joiv.2.1.100

Ramasubbu, N., & Kemerer, C. F. (2016). Technical debt and the reliability of enterprise software systems: A competing risks analysis. *Management Science*, *62*(5), 1487–1510. https://doi.org/10.1287/mnsc.2015.2196

Sarhan, Q. (2019). Best Practices and Recommendations for Writing Good Software. *The Journal of the University of Duhok*, *22*(1), 90–105. https://doi.org/10.26682/sjuod.2019.22.1.11

Verdecchia, R., Lago, P., Ebert, C., & De Vries, C. (2021). Green IT and Green Software. *IEEE Software*, *38*(6), 7–15. https://doi.org/10.1109/MS.2021.3102254