G-Tech : Jurnal Teknologi Terapan
Vol. 8, No. 1, Januari 2024, hlmn. 104-113
E-ISSN: 2623-064X | P-ISSN: 2580-8737

Analisis Strategi Algoritma *Sorting* Menggunakan Metode Komparatif pada Bahasa Pemrograman Java dengan Python

Joko Iskandar[1]✉, Heri Suhendar [2], Bian Dwi Pamungkas[3]

[1] Informatika, Universitas Bhinneka PGRI, Indonesia
[2] Teknik Informatika, Institut Teknologi Garut, Indonesia
[3] Pendidikan Teknologi Informasi, Universitas Bhinneka PGRI, Indonesia

Informasi Artikel

Riwayat Artikel

Diserahkan : 21-11-2023
Direvisi : 24-11-2023
Diterima : 02-12-2023

Kata Kunci:

algoritma sorting; java; pyhton

Keywords **:**

Java; python; sorthing algorithm

ABSTRAK

Penerapan strategi algoritma *sorting* yang tepat merupakan kunci penting yang tidak bisa dipandang kecil, khususnya bagi para pengembang sistem. Penelitian ini bertujuan menganalisis secara komprehensif dari enam strategi algoritma *sorting* dengan menggunakan dua bahasa pemrograman yaitu Java dan Python. Strategi algoritma *sorting* meliputi *bubble, selection*, *insertion, shell*, *merge* dan *quick sort*. Metode penelitiannya menggunakan pendekatan komparatif. Analisis datanya mulai dari 1000 data sampai dengan 200.000 data. Hasil penelitian menunjukkan bahwa **s**trategi algoritma *sort* untuk *bubble, selection* dan *insertion* pada jumlah lebih dari 20.000 data lebih cepat menggunakan bahasa Java dibanding dengan bahasa Python, sedangkan strategi *shell, merge* dan *quick sort* setelah lebih dari 20.000 menghasilkan waktu eksekusi yang tidak jauh berbeda baik yang menggunakan bahasa pemrograman Java maupun Python. Saat memilih bahasa pemrograman yang tepat untuk implementasi algoritma *sorting*, penting agar diperoleh sistem yang lebih efisien dalam penggunaan memori dan lebih cepat dari segi waktu.

ABSTRACT

Applying the right sorting algorithm strategy is a key factor that cannot be underestimated, especially for system developers. This research aims to comprehensively analyze six sorting algorithm strategies using two programming languages, Java and Python. The sorting algorithm strategies include bubble, selection, insertion, shell, merge and quick sort. The research method uses a comparative approach. The data analysis ranged from 1000 data to 200,000 data. The results showed that the sort algorithm strategies for bubble, selection and insertion at more than 20,000 data were faster using Java than Python, while the shell, merge and quick sort strategies after more than 20,000 resulted in execution times that were not much different using both Java and Python programming languages. When choosing the right programming language for the implementation of sorting algorithms, It is important to obtain a system that is more efficient in memory usage and faster in terms of time.

Corresponding Author **:**
Joko Iskandar
Informatika, Universitas Bhinneka PGRI, Indonesia
JL. Mayor Sujadi Timur No 7 Tulungagung, Kode Pos 66221
Email: arsip.indoscript@gmail.com

PENDAHULUAN

Algoritma merupakan suatu prosedur logis dalam pemecahan suatu masalah yang sangat penting yang diprioritaskan bagi setiap *developer*. Algoritma yang dirancang dan dipilih secara tidak tepat, dapat mengakibatkan proses eksekusi program dari segi waktu menjadi lama dan dari segi penggunaan memori yang banyak, bahkan tidak menutup kemungkinan menyebabkan kerusakan pada baik pada sistem maupun perangkat. Akibat dari pemilihan strategi algoritma yang tidak tidap tersebut dan program yang berjalan menjadi tdiak efisien menyebabkan sistem tidak bekerja secara maksimal serta pengguna sistem banyak yang mengeluh karena prosesnya yang lamban (Pratiwi, 2020).

Setiap program terutama berkaitan dengan sistem informasi akan berhubungan erat dengan penggunaan data atau pengolahan data dalam sistem tersebut. Semakin banyak data pada sistem, bahkan semakin banyak variasi peruntukan data pada sistem yang dibangun, akan menghasilkan pengolahan data menjadi tidak mudah dan masalahnya akan menjadi kompleks. Dalam proses tahapan pengolahan data dalam sistem yang umum digunakan adalah algoritma pencarian dan pengurutan. Algoritma pengurutan memerlukan strategi yang tepat agar data lebih mudah di baca dan di proses nantinya. Sampai saat ini, banyak strategi algoritma dalam pengurutan atau *sorting* yang umum digunakan oleh para programmer diantaranya adalah *bubble, selection, insertion, shell, merge* dan *quick sort* (Sunandar, 2019).

Strategi *bubble* dan *selection sort* memiliki keunggulan dalam mengurutkan data dengan jumlah sedikit. Sedangkan *insertion* merupakan algoritma pengurutan yang lebih cepat dibanding dengan *merge* dalam penggunaan data kurang dari 100, namun jika data lebih dari 100 ternyata *merge* lebih cepat dibanding dengan *insertion sort*. Namun *insertion* pada data yang lebih besar lagi ternyata masih lambat eksekusi waktu yang diperlukan dibanding dengan strategi *quick sort.* Dari sini *quick sort* menjadi strategi algoritma pengurutan yang lebih unggul pada data yang lebih besar (Heryanto & Harjanti, 2023).

Pada tahun 2019, Sunandar melakukan penelitian mengenai komparatif antara strategi algoritma pengurutan *selection* dan *insertion* menggunakan bahasa program Java. Strategi algoritma *selection* disusun dengan syntax yang lebih sederhana dibanding dengan *insertion* yang cenderung lebih rumit sehingga mengakibatkan waktu prosesnya menjadi lebih lama dibanding dengan *selection* untuk data kasus random (Sunandar, 2019). Namun berbeda dengan penelitian yang dilakukan oleh Saputra, dkk., tahun 2021 melakukan sebuah pengujian membandingkan algoritma pengurutan antara *bubble* dengan *selection* menunjukkan bahwa ternyata *selection* memiliki waktu eksekusinya lebih cepat pada jumlah data yang equal jika dibandingkan dengan algoritma *bubble* (Saputra dkk., 2021).

Pada tahun 2022, Sari, dkk., melakukan pengujian algoritma *selection* dan *insertion sort.* Hasil pengujian menunjukkan bahwa algoritma dalam pengurutan antara *selection* dan *insertion* yang paling efektif dan efisien ternyata adalah algoritma *selection* terutama dalam menangani pengolahan data untuk pengurutan (Sari, P. Y. dkk., 2022). Sementara Zulfa, dkk pada tahun 2022 yang sama telah melakukan pengujian algoritma pengurutan antara *bubble, shell*, dan *quick sort.* Pengujian dilakukan dengan bahasa pemrograman Java untuk mengurutkan baris angka secara acak. Hasil pengujiannya ternyata algoritma *quick* mempunyai performa yang lebih baik dibandingkan *bubble* dan *shell sort* dalam mengurutkan data secara acak (Zulfa dkk., 2022). Demikian juga dengan Tumanggor, dkk pada tahun 2022 melakukan penelitian dengan membandingkan *bubble*, algoritma *merge* dan algoritma *quick* didapatkan hasil bahwa algoritma *quick* dapat menyelesaikan secara rekursif sebanyak 25 kali dibanding *bubble* dan *merge* (Tumanggor dkk., 2022).

Pada penelitian terdahulu sudah banyak yang melakukan pengujian terhadap algoritma pengurutan untuk mendapatkan strategi dalam pengurutan data yang lebih cepat serta penggunaan memori yang lebih sedikit, namun pada penelitian terdahulu sebagian besar penelitian dilakukan secara parsial tidak komprehensif dan umumnya menggunakan bahasa pemrograman Java. Sehingga pada penelitian ini, penulis menganalisis enam strategi algoritma pengurutan data secara

komprehensif dan menggunakan dua bahasa pemrograman yang berbeda yaitu bahasa pemrograman Java dan Python. Hal ini bertujuan agar didapatkan hasil analisis perbandingan yang lengkap dari algoritma pengurutan data menggunakan dua bahasa pemrograman yang berbeda sehingga didapatkan algoritma dengan performa waktu proses lebih cepat serta penggunaan memori yang lebih sedikit. Manfaat dari penelitian ini memberikan informasi terhadap pemilihan algoritma *sorting* dalam implementasi pengembangan sebuah sistem sehingga didapatkan program yang lebih efisien dalam penggunaan memori dan lebih cepat dalam segi waktu.

METODE PENELITIAN

Metode penelitian yang digunakan yaitu penelitian komparatif. Penelitian komparatif merupakan penelitian yang membandingkan keberadaan satu variabel atau lebih pada dua atau lebih sampel yang berbeda, atau pada waktu yang berbeda (Sugiyono, 2018). Komparatif pada penelitian ini adalah membandingkan hasil pengujian sejumlah imputan data dari strategi algoritma pengurutan data (*sorting*) yaitu *bubble, selection, insertion, shell, merge* dan *quick sort* menggunakan dua bahasa pemrograman yang berbeda yaitu Java dan Python. Jumlah imputan data yang dianalisis dalam pengujian yaitu 1000, 2000, 3000, 5000, 10000, 20000, 50000, 100000 dan 200000.

Untuk melakukan pengujian strategi algoritma *sorting* menggunakan bahasa pemrograman Java dan Python, diperlukan perangkat meliputi *device* yaitu operasi sistemnya menggunakan Microsoft Windows 10 Pro dengan versi 10.0.19045 N/A dan build 19045, prosesor komputer menggunakan AMD Ryzen 5 tipe 3400G Radeon Vega Graphics 3.70 GHz, menggunakan memori 16,0 GB (13,9 GB *usable*) dengan tipe DDR4. Versi bahasa pemrograman untuk Java, menggunakan versi 11.0 dan Python menggunakan versi 3.11. Kemudian penulis membuat program masing-masing strategi algoritma *sorting* dengan pemrograman Java dan Python.

Pada tahap pengujian, dilakukan dengan melakukan aktifitas pengujian enam program strategi menggunakan dua bahasa pemrograman yang berbeda yaitu Java dan Python. Pertimbangan menggunakan kedua bahasa ini dikarenakan dua bahasa pemrograman baik Java maupun Python yang dua-duanya adalah bahasa pemrograman berbasis onjek yang saat ini masih banyak yang menggunakannnya. Sehingga hasil ini tentu akan sangat bermanfaat bagi para pengembang sistem khususnya dalam pemilihan strategi algoritma untuk pengurutan data. Hasil pengujian dengan menganalisis waktu esksekusi (t) dalam satuan detik (*second*) pada jumlah imputan data (n) yang telah ditentukan. Data dari hasil pengujian dilakukan analisis untuk mengetahui analisis komperatif antara bahasa pemrograman Java dan Python dalam mengeksekusi algoritma pengurutan data ke dalam bentuk tabel maupun grafik untuk mendapatkan gambaran yang lebih detail.

HASIL DAN PEMBAHASAN

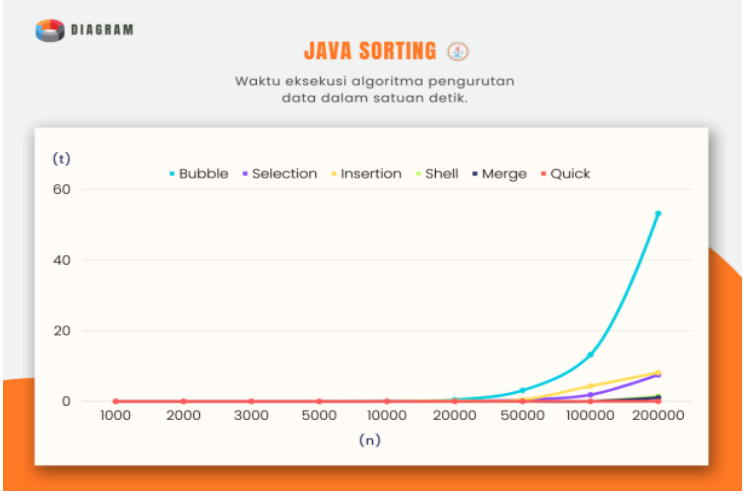
Hasil Pengujian Algoritma *Sorting* dengan Bahasa Pemrograman Java

Berikut ini adalah jumlah waktu yang dibutuhkan oleh strategi algoritma pengurutan data atau *sorting* menggunakan bahasa pemrograman Java yang dapat dilihat pada Tabel 1. Berdasarkan data pada Tabel 1, menunjukkan bahwa strategi algoritma *sorting* dengan pemrograman Java pada masukan data (n) mulai dari 1000 sampai dengan 20.000 tidak menghasilkan perbedaan yang bermakna, namun setelah masukan data lebih dari 20.000 hingga 200.000 mulai terlihat adanya perbedaan yang bermakna. Pada input data sebanyak 200.000 strategi algoritma *sorting* menggunakan bahasa pemrograman Java paling cepat dihasilkan oleh *quick sort* yang memerlukan waktu eksekusi (t) selama 0.025341 detik, sedangkan strategi yang paling lambat dihasilkan oleh *bubble sort* memerlukan waktu eksekusi (t) selama 53.24152 detik.

Tabel 1. Hasil Pengujian Strategi *Sorting* dengan Bahasa Pemrograman Java

Strategi / Jumlah data (n)	Waktu Eksekusi (t) dalam Satuan Detik (s)					
	Bubble Sort	*Selection Sort*	*Insertion Sort*	*Shell Sort*	*Merge Sort*	*Quick Sort*
1000	0.005237	0.002594	0.002154	0.000735	0.000793	0.000375
2000	0.009633	0.007526	0.003872	0.001691	0.000972	0.000493
3000	0.014684	0.010570	0.006970	0.001934	0.001204	0.000717
5000	0.029022	0.014097	0.015538	0.002880	0.001494	0.000898
10000	0.115845	0.030630	0.055320	0.005630	0.002963	0.001555
20000	0.473316	0.086414	0.182907	0.008212	0.006527	0.003068
50000	3.128752	0.477628	0.511289	0.011917	0.011610	0.007394
100000	13.23308	1.869878	4.334760	0.018568	0.019012	0.011080
200000	53.24152	7.527509	8.147127	1.534595	1.036998	0.025341

Untuk mendapatkan pemahaman yang lebih detail, ilustrasinya pada gambar berikut:



Gambar 1. Hasil Pengujian Strategi *Sorting* Pemrograman Java

Pada Gambar 1 menunjukkan bahwa grafik eksekusi pada strategi *bubble sort* (warna biru) semakin banyak data yang dibutuhkan akan semakin banyak pula waktu yang diperlukan untuk eksekusi, sementara untuk *quick sort* (warna merah) sampai jumlah 200.000 data hanya memerlukan waktu kurang dari 0.05 detik. Hasil penelitian menunjukkan bahwa strategi algoritma *sorting* menggunakan bahasa pemrograman Java yang paling cepat dihasilkan oleh *quick sort* dengan waktu eksekusi 0.025341 detik pada jumlah masukan sebanyak 200.000 data, sedangkan yang paling lambat dihasilkan *bubble sort* dengan waktu eksekusi 53.24152 detik pada jumlah data yang sama. Dari hasil analisis menunjukkan bahwa sampai dengan jumlah data sebanyak 20.000 data pada semua strategi *sort* baik *bubble*, *selection*, *shell*, *merge* maupun *quick sort* tidak menghasilan perbedaan yang bermakna, namun perbedaan terjadi setalah data yang dinputkan sejumlah 20.000 ke atas. Ini berarti bahwa untuk data yang jumlahnya tidak terlalu banyak maka strategi sort tidak punya pengaruh terhadap waktu eksekusi, akan tetapi jika data sangat banyak pada penelitian ini menunjukkan setelah data 20.000 yang dinputkan maka strategi *quick sort* adalah strategi sort yang cepat jika menggunakan bahasa pemrogramam Java.

Hasil penelian ini relevan dengan hasil penelitian Zulfa, dkk pada tahun 2022 yang sama telah melakukan pengujian algoritma pengurutan antara *bubble, shell*, dan *quick sort.* Pengujian dilakukan dengan bahasa pemrograman Java untuk mengurutkan baris angka secara acak. Hasil pengujiannya ternyata algoritma *quick* mempunyai performa yang lebih baik dibandingkan *bubble* dan *shell sort* dalam mengurutkan data secara acak (Zulfa dkk., 2022). Penelitian yang dilakukan oleh (Tambunan dkk., 2018) menggunakan bahasa pemrograman Java, menyatakan bahwa waktu eksekusi dalam *sorting* yang dibutuhkan menggunakan algoritma *insertion sort* dan *merge sort* menunjukkan kurang konsisten. Hasil penelitian yang dilakukan oleh (Anggreani dkk., 2020), menggunakan bahasa pemrograman Java, setelah melakukan implementasi dan analisis *merge sort*

memiliki waktu eksekusi yang lebih cepat dibandingkan dengan *insertion* dan *selection sort* dengan nilai rata-rata waktu eksekusi sebesar 108.593777 ms pada jumlah data 3000.

Penelitian Tumanggor, dkk yang melakukan penelitian dengan membandingkan *bubble sort*, algoritma *merge sort* dan algoritma *quick sort* didapatkan hasil bahwa algoritma *quick sort* dapat menyelesaikan secara rekursif sebanyak 25 kali dibanding *bubble sort* dan *merge sort* (Tumanggor dkk., 2022). *Sorting* merupakan operasi yang umum dilakukan dalam pemrograman komputer. Ini melibatkan pengurutan elemen-elemen dalam sebuah array atau struktur data sehingga mereka berada dalam urutan yang sesuai. Java menyediakan beberapa algoritma *sorting* yang berbeda, dan pemilihan algoritma yang tepat dapat memiliki dampak besar pada kinerja aplikasi. Hasil pengujian strategi algoritma *sorting* juga dapat dipengaruhi oleh spesifikasi komputer yang digunakan pada pengujian, dimana data inputan tercepat akan berbeda. Dari hasil penelitian ini, peneliti berasumsi bahwa apabila data yang diurutkan kurang dari 20.000 maka semua strategi dapat digunakan, tetapi apabila data lebih dari 20.000 maka perlu pemilihan dan strategi *quick sort* menunjukkan yang lebih efisien menggunakan bahasa pemrograman Java.

Hasil Pengujian Algoritma *Sorting* dengan Bahasa Pemrograman Python

Berdasarkan data hasil pengujian menunjukkan bahwa strategi algoritma *sorting* dengan menggunakan bahasa pemrograman Python pada masukan data (n) mulai dari 1000 sampai dengan 20.000 tidak menghasilkan perbedaan yang bermakna, namun setelah masukan data lebih dari 20.000 hingga 200.000 mulai terlihat adanya perbedaan yang bermakna. Pada masukan data (n) sebanyak 200.000 strategi algoritma *sorting* menggunakan bahasa pemrograman Python yang cepat dihasilkan oleh tiga strategi yaitu *shell sort* dengan waktu eksekusi 1.046303 detik, *merge sort* dengan waktu eksekusi 1.086408 detik dan *quick sort* dengan waktu eksekusi 0.086418 detik. Sedangkan strategi yang paling lambat dihasilkan oleh strategi algoritma pengurtan data *bubble sort* yang ternyata memerlukan waktu eksekusi 2657.128 detik. Data pengujian tersebut dapat dilihat pada Tabel 2.

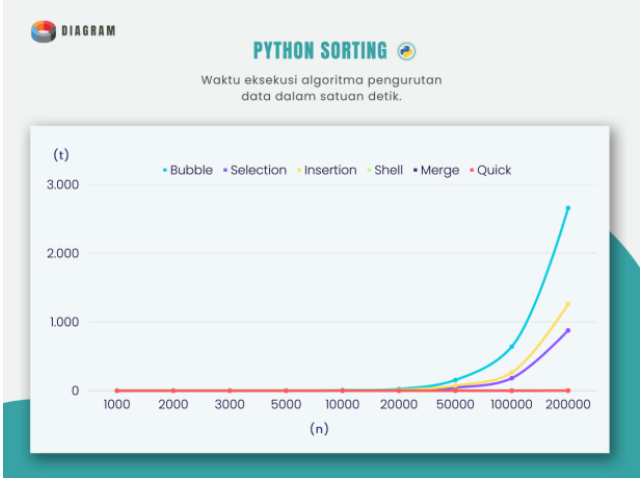
Tabel 2. Hasil Pengujian Strategi *Sorting* dengan Bahasa Pemrograman Python

Strategi / Jumlah data (n)	Waktu Eksekusi (t) dalam Satuan Detik (s)					
	Bubble Sort	*Selection Sort*	*Insertion Sort*	*Shell Sort*	*Merge Sort*	*Quick Sort*
1000	0.079866	0.020068	0.027088	0.002008	0.002005	0.001003
2000	0.227290	0.076506	0.099421	0.013921	0.005017	0.003008
3000	0.533590	0.17828	0.232786	0.010487	0.008021	0.005015
5000	1.710658	0.434436	0.632091	0.015048	0.013043	0.009022
10000	6.630937	1.743769	2.585544	0.033110	0.027089	0.020067
20000	24.83016	7.076412	11.52913	0.075249	0.060199	0.038126
50000	155.1172	44.83533	74.40416	0.229762	0.189630	0.157517
100000	641.3104	182.2168	264.2382	0.409344	0.350125	0.345175
200000	2657.128	877.5564	1261.145	1.046303	1.086408	0.086418

Pada Tabel 2 maupun pada Gambar 14 berikut ini terlihat bahwa *bubble sort* memerlukan waktu yang lama ketika data betambah lebih banyak. Pada Gambar 2, grafik waktu eksekusi pada strategi *bubble sort* (warna biru) semakin banyak data yang dibutuhkan akan semakin banyak waktu yang diperlukan untuk eksekusi, sementara ada tiga strategi yang memerlukan waktu cepat untuk mengeksekusi data hingga 200.000 data yaitu *shell sort* dengan waktu 1.046303 detik, *merge sort* dengan waktu 1.086408 detik dan *quick sort* dengan waktu 0.086418 detik.

Hasil penelitian menunjukkan bahwa strategi algoritma *sorting* paling cepat menggunakan bahasa pemrograman Python untuk jumlah 20.000 sampai dengan 200.000 data dihasilkan oleh tiga strategi yaitu *shell sort* dengan waktu eksekusi 1.046303 detik, *merge sort* dengan waktu eksekusi 1.086408 detik dan *quick sort* dengan waktu eksekusi 0.086418 detik, sedangkan strategi yang paling lambat dihasilkan oleh *bubble sort* dengan waktu eksekusi 2657.128 detik. Perbedaan dengan bahasa pemrograman Java, terlihat setelah data sebanyak 20.000. Jika data relatif kecil atau masih di bawah 20.000 maka untuk semua strategi algoritma pengurutan data menggunakan bahasa pemrograman Pyhton tidak menunjukkan perbedaan yang bermakna. Namun, setalah 20.000 ke

atas jika menggunakan Python, maka ada tiga strategi yang dapat digunakan untuk pengurutan data yaitu *shell, merge* dan *quick sort.*

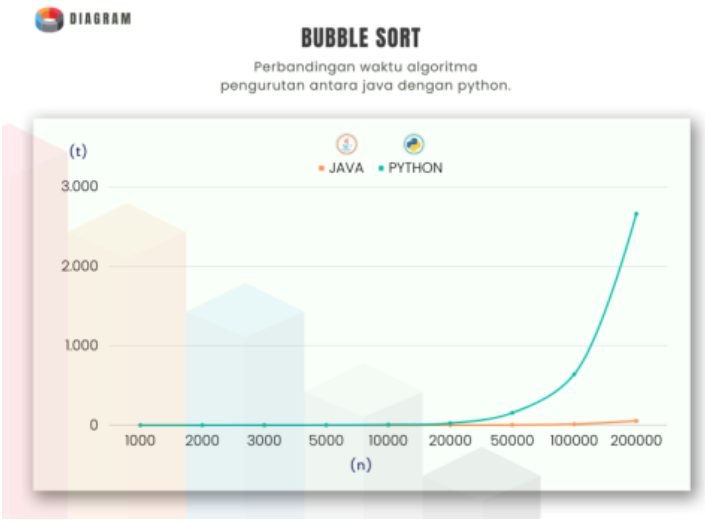


Gambar 2. Hasil Pengujian Strategi *Sorting* Pemrograman Python

Hasil penelitian ini mendukung penelitian yang dilakukan oleh (Heryanto & Harjanti, 2023) yang menggunakan bahasa pemrograman Python, didapatkan hasil bahwa algoritma *quick sort* lebih cepat pada semua inputan yang dianalisis mulai dari 200 sampai 500 dengan waktu eksekusi kurang dari 0,003 detik. Juga penelitian yang dilakukan oleh (Saputra dkk., 2021) yang membandingkan antara *bubble* dan *selection sort*, menunjukkan bahwa selection sort dinyatakan pemilik waktu eksekusi tercepat dibandingkan dengan *bubble sort*.

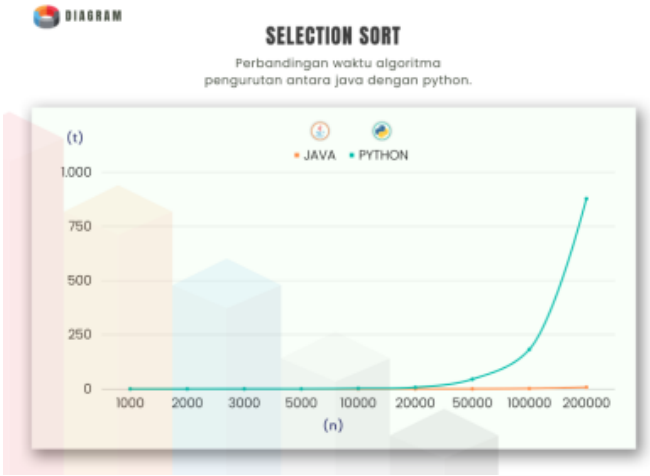
Hasil Perbandingan Algoritma *Sorting* menggunakan Bahasa Pemrograman Java dan Python

Setelah didapatkan hasil pengujian antara algoritma sorting menggunakan bahasa pemrograman Java dan Python pada semua inputan data, maka selanjutkan dilakukan perbandingan antara kedua hasil pengujian tersebut. Berikut ini adalah perbandingan algoritma sorting menggunakan bahasa pemrograman Java dan Python yang dapat dilihat pada gambar 15.



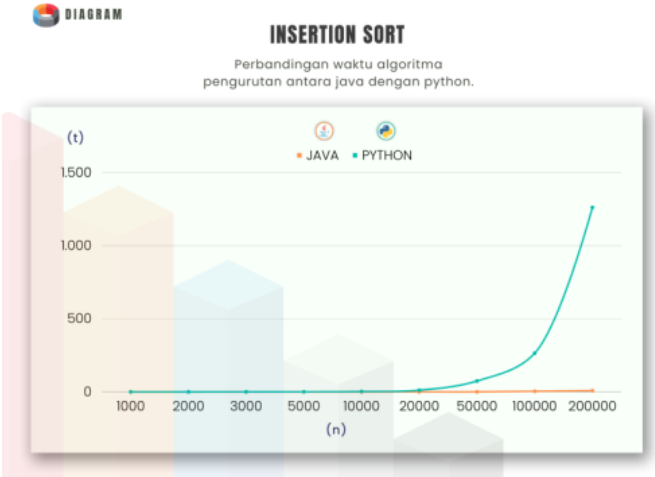
Gambar 3. *Bubble Sort* Java dan Python

Pada Gambar 3 menunjukkan bahwa algoritma *bubble sort* antara bahasa pemrograman Java dan Python sampai jumlah data 20.000 hampir sama, perbedaan waktu eksekusi terjadi setelah masukan data lebih dari 50.000 dan setelah 200.000 data waktu eksekusi dengan bahasa pemrograman Java untuk strategi *bubble sort* lebih cepat dibanding menggunakan bahasa pemrograman Python.

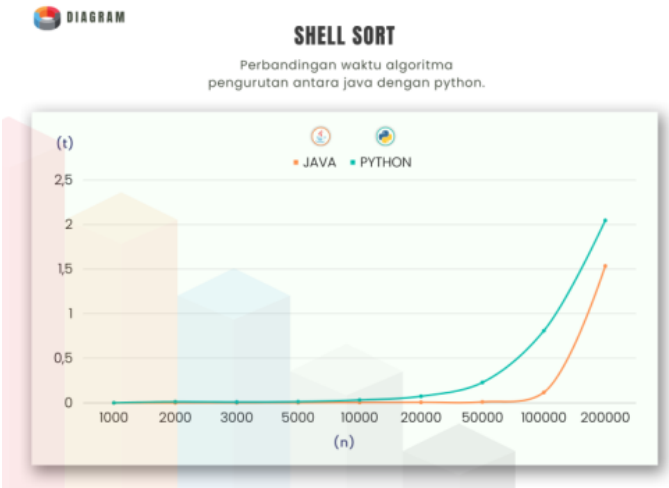


Gambar 4. *Selection Sort* Java dan Python

Pada Gambar 4, menunjukkan bahwa algoritma *selection sort* dengan pemrograman Java dan Python sampai jumlah data 20.000 hampir sama, perbedaan waktu eksekusi terjadi setelah masukan data lebih dari 50.000 dan setelah 200.000 data waktu eksekusi dengan bahasa pemrograman Java untuk strategi *selection sort* lebih cepat dibanding menggunakan bahasa pemrograman Python.

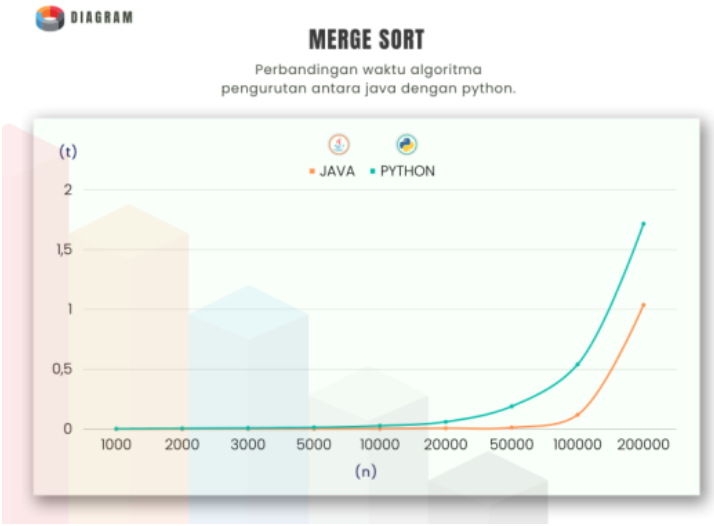


Gambar 5. *Insertion Sort* Java dan Python



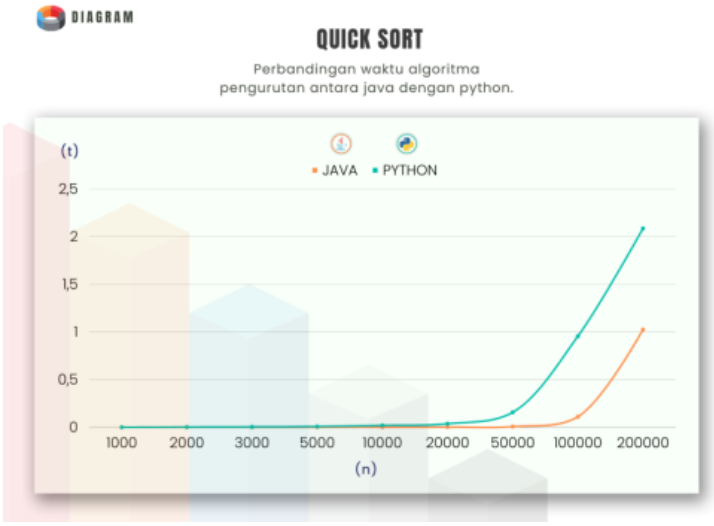
Gambar 6. *Quick Sort* Java dan Python

Pada Gambar 5 menunjukkan bahwa strategi algoritma *insertion sort* antara bahasa pemrograman Java dan Python sampai jumlah data 20.000 hampir sama, perbedaan waktu eksekusi terjadi setelah masukan data lebih dari 50.000 dan setelah 200.000 data waktu eksekusi dengan bahasa pemrograman Java untuk strategi *insertion sort* lebih cepat dibanding menggunakan bahasa pemrograman Python. Pada Gambar 6 diatas menunjukkan bahwa strategi algoritma *shell sort* antara bahasa pemrograman Java dan bahasa pemrograman Python waktu eksekusi yang diperlukan hampir sama, yaitu setelah lebih dari 20.000 data yang dinputkan memerlukan waktu yang hampir sama *shell sort* menggunakan bahasa pemrograman Java dan Python.



Gambar 7. *Merge Sort* Java dan Python

Pada Gambar 7 diatas menunjukkan bahwa strategi algoritma *merge sort* antara bahasa pemrograman Java dan bahasa pemrograman Python waktu eksekusi yang diperlukan hampir sama, yaitu setelah lebih dari 20.000 data yang dinputkan memerlukan waktu yang hampir sama *merge sort* menggunakan bahasa pemrograman Java dan Python. Pada Gambar 8, strategi algoritma *quick sort* antara bahasa pemrograman Java dan Python waktu eksekusi yang diperlukan hampir sama, yaitu setelah lebih dari 20.000 memerlukan waktu yang meningkat *quick sort* menggunakan bahasa pemrograman Java dan Python. Hasil penelitian menunjukkan bahwa strategi algoritma *bubble sort, selection sort* dan *insertion sort* untuk jumlah lebih dari 20.000 data lebih cepat menggunakan bahasa Java dibanding dengan bahasa Python, sedangkan strategi *shell sort, merge sort* dan *quick sort* setelah lebih dari 20.000 memerlukan waktu yang tidak jauh berbeda antara Java maupun Python.



Gambar 8. *Merge Sort* Java dan Python

Temuan ini sejalan dengan hasil penelitian yang dilakukan oleh (Sari, N. dkk., 2022), yang menyatakan bahwa bubble sort menunjukkan kinerja pengurutan yang kurang efisien dalam bahasa pemrograman Java, terutama ketika menangani dataset yang besar. Dalam konteks bahasa pemrograman Java, strategi bubble sort dapat mengalami penurunan signifikan dalam kecepatan pengurutan ketika berurusan dengan volume data yang sangat besar. Temuan ini mendukung penelitian sebelumnya yang dilakukan oleh Retnoningsih, yang menyatakan bahwa tidak ada algoritma yang secara universal terbaik untuk semua situasi. Sebaliknya, efisiensi suatu algoritma dapat tergantung pada jumlah data yang diaturnya. Hasil uji perbandingan antara insertion sort dan selection sort yang dilakukan oleh Retnoningsih pada tahun 2018 menunjukkan bahwa insertion sort lebih unggul dalam pengurutan jumlah data yang sedikit, sementara selection sort memiliki kinerja lebih baik ketika jumlah data lebih banyak atau meningkat (Retnoningsih, 2018).

Implementasi algoritma sorting, bahasa pemrograman Python cenderung lebih pendek dan lebih mudah dibaca dibandingkan Java, terutama karena Python memiliki sintaksis yang lebih ringkas. Namun, dalam hal performa, Java cenderung lebih cepat karena kode Java dikompilasi menjadi bytecode yang dijalankan oleh mesin virtual Java yang sangat optimal. Saat memilih bahasa pemrograman yang tepat untuk implementasi algoritma sorting, penting untuk mempertimbangkan pemilihan strategi yang sesuai dengan kebutuhan proyek, preferensi tim, dan aspek kinerja, sehingga sistem yang dibangun lebih efisien dalam penggunaan memori dan lebih cepat dari segi waktu.

KESIMPULAN DAN SARAN

Kesimpulan

Berdasarkan hasil penelitian ini, maka dapat diambil kesimpulan sebagai berikut yaitu untuk strategi algoritma *sorting* paling cepat menggunakan bahasa pemrograman Java untuk jumlah 20.000 sampai dengan 200.000 data dihasilkan oleh *quick sort*, sedangkan yang paling lambat dihasilkan *bubble sort*. Strategi algoritma *sorting* paling cepat menggunakan bahasa pemrograman Python untuk jumlah 20.000 sampai dengan 200.000 data dihasilkan oleh tiga strategi yaitu *shell sort*, *merge sort* dan *quick sort*, sedangkan strategi yang paling lambat dihasilkan oleh *bubble sort*. Hasil penelitian ini menemukan bahwa strategi algoritma *bubble sort, selection sort* dan *insertion sort* untuk jumlah lebih dari 20.000 data lebih cepat menggunakan bahasa Java dibanding dengan bahasa Python, sedangkan strategi *shell sort, merge sort* dan *quick sort* setelah lebih dari 20.000 memerlukan waktu yang tidak jauh berbeda yang menggunakan bahasa pemrograman Java maupun Python.

Saran

Berdasarkan hasil penelitian ini diharapkan dapat memberikan masukan kepada pengembang sistem mengenai strategi *sorting* yang efektif dan efisien sehingga menghasilkan sistem yang dapat diakses oleh pengguna secara cepat dan tepat, serta memberikan dasar bagi penelitian yang akan datang terkait dengan algoritma.

UCAPAN TERIMA KASIH

Kami mengucapkan terima kasih kepada Rektor Universitas Bhinneka PGRI dan Kepala LPPM Universitas Bhinneka PGRI yang telah menyelenggarakan program hibah penelitian internal, serta kepada rekan-rekan dosen Informatika atas dukungan dan supportnya.

REFERENSI

Anggreani, D., Wibawa, A. P., Purnawansyah, P., & Herman, H. (2020). Perbandingan Efisiensi Algoritma Sorting dalam Penggunaan Bandwidth. *ILKOM Jurnal Ilmiah*, *12*(2), 96–103. https://doi.org/10.33096/ilkom.v12i2.538.96-103

Heryanto, Y., & Harjanti, T. W. (2023). Analisis Perbandingan Ruang dan Waktu pada Algoritma Sorting Menggunakan Bahasa Pemrograman Python. *Jurnal Penerapan Sistem Informasi (Komputer & Manajemen)*, *4*(2), 342–347.

Pratiwi, E. L. (2020). *Konsep Dasar Algoritma dan Pemrograman dengan Bahasa Java*. Banjarmasin: Poiban Press.

Retnoningsih, E. (2018). Algoritma Pengurutan Data (Sorting) dengan Metode Insertion Sort dan Selection Sort. *Information Management for Educators and Professionals*, *3*(1), 95–106.

Saputra, D. Y. R., Andryana, S., & Sholihati, I. D. (2021). Analisis Perbandingan Algoritma Bubble Sort Dan Selection Sort Pada Sistem Pendukung Keputusan Pemilihan Tempat Kost Berbasis Ios (Iphone Operating System). *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)*, *6*(2), 318–324. https://doi.org/10.29100/jipi.v6i2.2015

Sari, N., Gunawan, W. A., Sari, P. K., Zikri, I., & Syahputra, A. (2022). Analisis Algoritma Bubble Sort Secara Ascending dan Descending Serta Implementasinya dengan Menggunakan Bahasa Pemrograman Java. *ADI Bisnis Digital Interdisiplin Jurnal*, *3*(1), 16–23. https://doi.org/10.34306/abdi.v3i1.625

Sari, P. Y., Ali, R., & Rajasa, A. (2022). Perbandingan Efisiensi dengan Algoritma Sorting dalam Penentuan Jarak (Studi Kasus: Pet Shop di Bandar Lampung). *Jurnal TEKNIKA POLSRI*, *35142*(93), 149–159.

Sugiyono. (2018). *Metode Penelitian Kuantitatif*. Bandung: Alfabeta.

Sunandar, E. (2019). Perbandingan Metode Selection Sort dan Insertion Sort Dalam Pengurutan Data Menggunakan Bahasa Program Java. *Petir*, *12*(2), 172–178. https://doi.org/10.33322/petir.v12i2.485

Tambunan, H. satria, Sumarno, Giunawan, I., & Irawan, E. (2018). Optimasi Algoritma Shell Sort Dalam Pengurutan Data Huruf dan Angka. *Junal Sistem Informasi Ilmu Komputer Prima*, *2*(1), 23–27.

Tumanggor, Y., Maya, R., Lubis, F., Sianturi, M. P., & Purba, R. G. (2022). Metode Algoritma Bubble Sort , Algoritma Merge Sort dan Algoritma Quick Sort Dalam Pengujian Perbandingan Proses Penelitian Kualitatif. *JUTISAL (Jurnal Teknik Informatika Komputer Universal)*, *2*(2), 47–58.

Zulfa, M. L., Mikhael, & Sari, B. N. (2022). Analisis Perbandingan Algoritma Bubble Sort, Shell Sort, dan Quick Sort dalam Mengurutkan Baris Angka Acak menggunakan Bahasa Java. *Jurnal Ilmiah Wahana Pendidikan*, *2022*(13), 237–246. Diambil dari https://doi.org/10.5281/zenodo.6962346