Penerapan Algoritma Program Dinamis dalam Manajemen Rutinitas Harian

Mohammad Nugraha Eka Prawira - 13522001
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
mnugrahaekaprawira@gmail.com

Abstrak— Penjadwalan tugas harian merupakan aspek kunci dalam produktivitas sehari-hari. Dalam dunia yang penuh dengan tuntutan dan keterbatasan waktu, memiliki rencana yang efektif untuk menyelesaikan tugas-tugas menjadi sangat penting. Dalam makalah ini, kami mengusulkan pembuatan program dinamis untuk membantu pengguna dalam menyusun rencana terbaik untuk penjadwalan tugas harian mereka. Kami memperkenalkan konsep algoritma dinamis dan mengimplementasikannya dalam bahasa pemrograman Java untuk mencapai tujuan tersebut.

Kata kunci— Java, tugas harian, algoritma, program dinamis

***Abstract*— Daily task scheduling is a key aspect of everyday productivity. In a world filled with demands and time constraints, having an effective plan to accomplish tasks becomes crucial. In this paper, we propose the development of a dynamic program to assist users in creating the best plan for scheduling their daily tasks. We introduce the concept of a dynamic algorithm and implement it in the Java programming language to achieve this goal.**

Key word— Java, daily tasks, algorithm, dynamic program

1. PENDAHULUAN

Penjadwalan tugas harian sering kali merupakan tantangan bagi banyak orang. Dengan banyaknya tugas yang harus diselesaikan dan waktu yang terbatas, penting bagi kita untuk memiliki alat yang dapat membantu menyusun rencana yang efisien. Dalam makalah ini, saya mengajukan pendekatan menggunakan algoritma dinamis untuk menyelesaikan masalah penjadwalan tugas harian. Saya akan membahas konsep dasar algoritma dinamis, kemudian saya akan mengimplementasikan program sederhana dalam bahasa Java yang dapat membantu pengguna dalam menyusun rencana terbaik berdasarkan prioritas dan durasi tugas.

Rutinitas harian yang baik dapat meningkatkan produktivitas, mengurangi stres, dan membantu individu mencapai tujuan mereka dengan lebih efisien. Namun, menyusun rencana harian yang optimal tidak selalu mudah. Faktor-faktor seperti banyaknya tugas, durasi setiap tugas, prioritas, dan keterbatasan waktu membuat penjadwalan menjadi kompleks. Tanpa alat bantu yang tepat, seseorang mungkin kesulitan menentukan urutan tugas yang paling efisien.

Algoritma program dinamis adalah metode yang sangat efisien untuk menangani masalah optimasi seperti ini. Teknik ini memungkinkan kita untuk memecah masalah besar menjadi bagian-bagian yang lebih kecil dan menyelesaikannya secara berulang-ulang, sambil menyimpan hasil dari sub-masalah yang telah diselesaikan. Dengan cara ini, algoritma program dinamis dapat menghindari perhitungan yang berulang dan mengurangi waktu komputasi secara signifikan.

Dalam makalah ini akan dibahas implementasi algoritma program dinamis untuk penjadwalan tugas harian. Akan diperkenalkan konsep dasar dari program dinamis, menjelaskan bagaimana algoritma ini dapat digunakan untuk menyusun rencana harian yang optimal, dan memberikan contoh implementasi dalam bahasa pemrograman Java.

Makalah ini diharapkan dapat memberikan wawasan dan panduan praktis bagi mereka yang ingin menggunakan teknologi untuk meningkatkan produktivitas pribadi melalui penjadwalan yang lebih baik. Selain itu, implementasi program yang diusulkan juga dapat diadaptasi untuk berbagai keperluan lainnya dalam manajemen waktu dan tugas.

2. DASAR TEORI

2.1. PROGRAM DINAMIS

Program Dinamis (Dynamic Programming) adalah teknik optimisasi yang digunakan untuk menyelesaikan masalah kompleks dengan membaginya menjadi sub-masalah yang lebih sederhana. Teknik ini sering diterapkan dalam berbagai disiplin ilmu, termasuk matematika, ilmu komputer,

ekonomi, dan manajemen, karena kemampuannya untuk mengurangi redundansi komputasi melalui penyimpanan hasil sub-masalah yang telah diselesaikan.

2.1.1. Konsep Dasar Program Dinamis

Program dinamis beroperasi berdasarkan prinsip optimisasi "bottom-up" dan teknik "memoisasi". Berikut adalah konsep-konsep kunci yang mendasari program dinamis:

a. **Dekomposisi Masalah:** Masalah utama dipecah menjadi sub-masalah yang lebih kecil dan memiliki struktur serupa dengan masalah asli. Setiap sub-masalah diselesaikan secara independen, dan hasilnya digunakan untuk menyelesaikan masalah yang lebih besar.

b. **Memoisasi (Memoization):** Memoisasi adalah teknik penyimpanan hasil dari sub-masalah yang telah diselesaikan untuk menghindari perhitungan ulang saat sub-masalah tersebut dibutuhkan kembali. Hasil-hasil ini biasanya disimpan dalam tabel atau array.

c. **Relasi Rekursif:** Relasi rekursif mendefinisikan bagaimana solusi dari sub-masalah dapat digabungkan untuk membentuk solusi dari masalah yang lebih besar. Relasi ini menjadi dasar untuk memecahkan masalah secara bertahap.

d. **Optimal Substructure:** Masalah memiliki struktur optimal jika solusi optimal untuk keseluruhan masalah dapat dibangun dari solusi optimal untuk sub-masalahnya.

e. **Overlapping Subproblems:** Banyak sub-masalah yang sama dihitung berulang kali dalam proses pemecahan masalah. Program dinamis mengatasi ini dengan menyimpan hasil sub-masalah yang telah dihitung.

2.1.2. Langkah-langkah Program Dinamis

Penerapan program dinamis melibatkan beberapa langkah sistematis:

a. **Identifikasi Struktur Optimal Subproblems:** Menentukan bagaimana masalah dapat dipecah menjadi sub-masalah dan bagaimana solusi dari sub-masalah dapat digabungkan untuk membentuk solusi keseluruhan.

b. **Definisikan Relasi Rekursif:** Membuat rumusan yang menunjukkan bagaimana solusi dari sub-masalah digunakan untuk membentuk solusi dari masalah yang lebih besar.

c. **Hitung Nilai Sub-Masalah:** Memulai dari sub-masalah terkecil dan menghitung nilai sub-masalah secara iteratif hingga mencapai solusi masalah utama.

d. **Memoisasi Hasil Sub-Masalah:** Menyimpan hasil setiap sub-masalah dalam tabel atau array untuk menghindari perhitungan ulang.

e. **Konstruksi Solusi Optimal:** Menggunakan nilai yang disimpan untuk membangun solusi optimal untuk masalah utama.

2.1.3. Contoh Penerapan Program Dinamis

Berikut adalah beberapa contoh klasik dari penerapan program dinamis:

a. **Masalah Fibonacci:**

$$F(n) = \begin{cases} 0 & \text{Untuk n =1} \\ 1 & \text{Untuk n =2} \\ F(n-1) + F(n-2) & \text{Untuk n >2} \end{cases}$$

Dengan program dinamis, kita dapat menyimpan hasil perhitungan sebelumnya untuk menghindari perhitungan ulang.

b. **Masalah Knapsack:** Diberikan sejumlah item dengan berat dan nilai tertentu, dan sebuah tas dengan kapasitas maksimum, tentukan kombinasi item yang dapat dimasukkan ke dalam tas sehingga nilai total maksimal. Sub-masalah: Memilih item untuk mencapai kapasitas tertentu tanpa melampaui batas.

c. **Masalah Jalur Terpendek (Shortest Path):** Dalam graf berarah dengan bobot, temukan jalur terpendek dari satu simpul ke simpul lainnya. Algoritma seperti Floyd-Warshall menggunakan program dinamis untuk menghitung jarak terpendek antara semua pasangan simpul.

d. **Masalah Pengurutan Terkait (Longest Common Subsequence):** Menemukan subsekuens terpanjang yang umum dari dua urutan. Sub-masalah: Menghitung subsekuens

terpanjang untuk bagian awal dari dua urutan.

2.2. Manajemen Waktu dan Penjadwalan Kegiatan

Manajemen waktu adalah proses perencanaan dan pengendalian berapa banyak waktu yang dihabiskan untuk kegiatan tertentu. Penjadwalan adalah salah satu alat utama dalam manajemen waktu yang membantu individu atau organisasi dalam mengalokasikan waktu secara efektif untuk berbagai kegiatan. Dalam konteks penjadwalan tugas harian, penting untuk memiliki pendekatan yang sistematis untuk mengatur dan menyusun urutan kegiatan berdasarkan durasi dan prioritasnya.

2.3. Pentingnya Manajemen Rutinitas Harian

Rutinitas harian yang baik dapat meningkatkan efisiensi dan produktivitas, mengurangi stres, dan membantu mencapai tujuan dengan lebih efektif. Manajemen rutinitas harian melibatkan identifikasi tugas-tugas yang perlu diselesaikan, menentukan durasi yang dibutuhkan untuk setiap tugas, dan menetapkan prioritas untuk masing-masing tugas. Penjadwalan yang efektif memerlukan alat dan teknik yang dapat membantu dalam mengatur dan mengoptimalkan urutan kegiatan.

2.4. Teori Dasar Penjadwalan Kegiatan

Penjadwalan kegiatan berdasarkan durasi dan prioritas melibatkan beberapa konsep dasar yang perlu dipahami:

a. **Durasi Kegiatan**: Durasi adalah waktu yang diperlukan untuk menyelesaikan suatu tugas atau kegiatan. Penjadwalan yang baik memerlukan estimasi yang akurat dari durasi masing-masing tugas untuk mengalokasikan waktu secara efektif.

b. **Prioritas Kegiatan**: Prioritas adalah tingkat kepentingan atau urgensi dari suatu tugas. Tugas dengan prioritas lebih tinggi biasanya harus diselesaikan terlebih dahulu dibandingkan dengan tugas-tugas dengan prioritas lebih rendah.

c. **Optimalisasi Jadwal**: Optimalisasi jadwal melibatkan penyusunan urutan kegiatan sedemikian rupa sehingga tugas-tugas dengan prioritas tinggi diselesaikan lebih dulu, sambil memastikan bahwa semua tugas dapat diselesaikan dalam batasan waktu yang tersedia

3. IMPLEMENTASI DAN PENGUJIAN

Pada bagian ini, akan dibahas implementasi program dinamis untuk penjadwalan tugas harian dalam bahasa pemrograman Java. Program ini mengumpulkan input tugas dari pengguna, menghitung jadwal optimal berdasarkan durasi dan prioritas tugas, serta menampilkan rencana terbaik yang dapat diselesaikan dalam waktu yang tersedia. Nilai prioritas yang lebih besar menunjukkan tingkat prioritas yang lebih tinggi. Artinya, tugas dengan nilai prioritas yang lebih besar dianggap lebih penting atau mendesak dan akan diprioritaskan lebih tinggi dalam penjadwalan. Implementasi algoritma dinamis dalam kode memastikan bahwa tugas dengan prioritas yang lebih tinggi diutamakan selama masih dalam batasan waktu yang tersedia.

Berikut merupakan code yang mempunyai tiga parameter yaitu nama kegiatan, durasi kegiatan, dan prioritas kegitan untuk menentukan urutan kegiatan apa saja yang harus dilakukan beserta urutan kegiatannya sesuai dengan prioritas kegiatan tersebut.

Gambar 3.1 Code Program Keseluruhan

3.1. Pendefinisian Kelas Task:

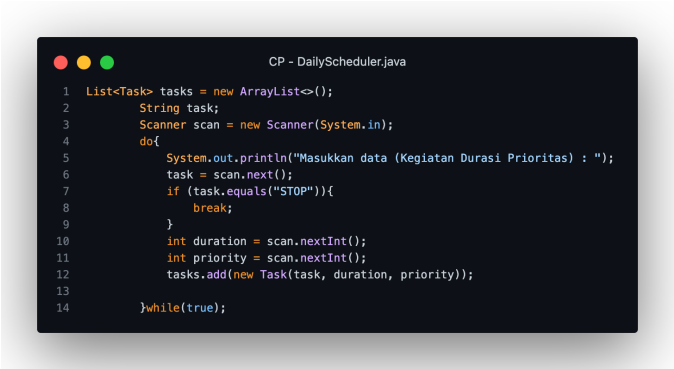
```java
class Task {
    String name;
    int duration;
    int priority;

    public Task(String name, int duration, int priority) {
        this.name = name;
        this.duration = duration;
        this.priority = priority;
    }
}
```

Gambar 3.2 Kelas Task

Kelas Task digunakan untuk menyimpan informasi mengenai setiap tugas, termasuk nama tugas (name), durasi (duration), dan prioritas (priority). Kelas ini memiliki konstruktor untuk menginisialisasi objek Task.

3.2. Pengumpulan Input Tugas



Gambar 3.3 Code Untuk Penerimaan Input dari User

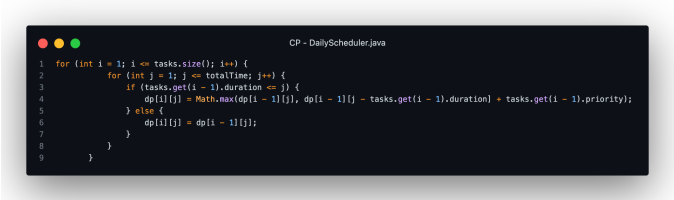
Bagian ini meminta pengguna untuk memasukkan data tugas, yang meliputi nama tugas, durasi, dan prioritas. Pengguna dapat memasukkan "STOP" untuk menghentikan proses input. Data tugas disimpan dalam daftar tasks.

3.3. Inisialisasi Tabel DP

Gambar 3.4 Code Untuk Penerimaan Input Waktu dari User

Tabel dp digunakan untuk menyimpan hasil perhitungan sub-masalah. dp[i][j] menyimpan nilai prioritas maksimum yang dapat diperoleh dengan mempertimbangkan i tugas pertama dalam j jam.

3.4. Perhitungan Program Dinamis



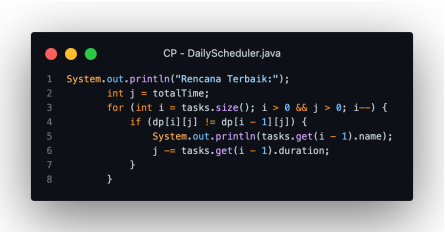
Gambar 3.5 Perhitungan Penentuan Prioritas Kegiatan

Bagian ini mengisi tabel dp dengan menggunakan prinsip program dinamis. Jika durasi tugas saat ini tidak melebihi waktu yang tersedia (j), maka nilai maksimum antara menyertakan tugas saat ini atau tidak menyertakannya akan diambil.

a. **Pengecekan Durasi Tugas**: Jika durasi tugas (tasks.get(i - 1).duration) lebih kecil atau sama dengan waktu yang tersedia (j), kita akan menghitung nilai maksimum antara tidak menyertakan tugas saat ini (dp[i - 1][j]) atau menyertakan tugas saat ini (dp[i - 1][j - tasks.get(i - 1).duration] + tasks.get(i - 1).priority).

b. **Tidak Menyertakan Tugas**: Jika durasi tugas lebih besar dari waktu yang tersedia, maka kita hanya mempertahankan nilai dari dp[i - 1][j].

3.5. Konstruksi dan Tampilan Solusi Optimal



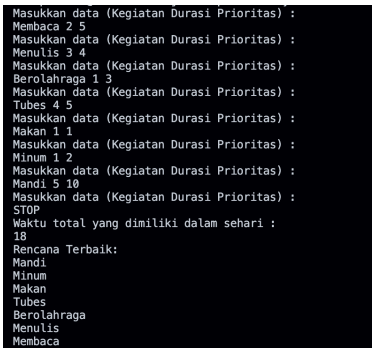
Gambar 3.6 Mengeluarkan Output Rencana

Bagian ini merekonstruksi dan menampilkan rencana terbaik berdasarkan nilai-nilai yang tersimpan dalam tabel dp. Tugas yang berkontribusi terhadap solusi optimal akan ditampilkan.

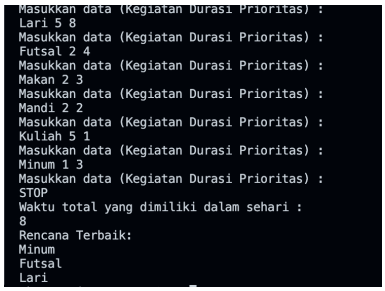
a. **Iterasi Balik**: Dengan mengiterasi balik dari jumlah tugas (tasks.size()) dan waktu total (totalTime), kita dapat menentukan tugas mana saja yang termasuk dalam solusi optimal.

b. **Pengecekan Solusi Optimal**: Jika nilai pada dp[i][j] berbeda dengan nilai pada dp[i - 1][j], berarti tugas tersebut dimasukkan dalam rencana. Nama tugas tersebut kemudian dicetak, dan waktu yang tersisa (j) dikurangi dengan durasi tugas tersebut.

3.6. Pengujian



Gambar 3.7 Pengujian Pertama



Gambar 3.7 Pengujian Pertama

Dengan pendekatan ini, penjadwalan tugas harian dapat dilakukan dengan lebih efektif dan efisien, memungkinkan pengguna untuk memaksimalkan produktivitas dan mengelola waktu dengan lebih baik. Algoritma program dinamis memberikan solusi yang adaptif dan optimal dalam menghadapi berbagai macam tugas dan prioritas.

4. ANALISIS

4.1. Efisiensi Algoritma

Implementasi algoritma dinamis dalam penjadwalan tugas harian menunjukkan beberapa keuntungan dan tantangan yang penting untuk dianalisis lebih lanjut.

a. Kompleksitas Waktu:

- Algoritma ini memiliki kompleksitas waktu O(nW), di mana n adalah jumlah tugas dan W adalah waktu total yang tersedia (dalam satuan jam). Kompleksitas ini berasal dari pengisian tabel dp yang berukuran (n+1) x (W+1).
- Dalam skenario praktis, kompleksitas ini cukup efisien untuk jumlah tugas dan durasi yang realistis dalam penjadwalan harian, mengingat batasan waktu biasanya tidak terlalu besar (misalnya, 8-12 jam).

b. Kompleksitas Ruang:

- Algoritma menggunakan tabel dp berukuran (n+1) x (W+1), yang berarti kompleksitas ruangnya adalah O(nW). Hal ini bisa menjadi kendala jika jumlah tugas (n) atau total waktu (W) sangat besar.
- Namun, dalam konteks penjadwalan harian dengan jumlah tugas yang terbatas, penggunaan memori ini masih dalam batas yang dapat diterima.

c. Optimalitas Solusi:

- Algoritma dinamis memastikan bahwa solusi yang dihasilkan adalah optimal dengan memanfaatkan prinsip substruktur optimal dan penyimpanan hasil sub-masalah (memoisasi).
- Ini berarti solusi yang ditemukan akan memaksimalkan prioritas total dari tugas yang dapat diselesaikan dalam waktu yang tersedia.

5. Keuntungan dan Keterbatasan

5.1. Keuntungan:

a. Optimalisasi Produktivitas: Dengan memprioritaskan tugas berdasarkan durasi dan prioritas, pengguna dapat memastikan bahwa waktu mereka digunakan dengan cara yang paling produktif.

b. Adaptabilitas: Algoritma ini dapat diadaptasi untuk berbagai skenario penjadwalan, tidak hanya untuk tugas harian tetapi juga untuk penjadwalan proyek atau tugas mingguan.

c. Kesederhanaan Implementasi: Meskipun menggunakan konsep yang canggih, implementasi dalam bahasa Java cukup sederhana dan mudah dimengerti, memungkinkan pengguna dengan pengetahuan pemrograman dasar untuk menggunakannya.

5.2. Keterbatasan:

a. Prioritas Subjektif: Penetapan prioritas tugas bersifat subjektif dan bergantung pada penilaian pengguna. Jika prioritas tidak ditetapkan dengan benar, solusi yang dihasilkan mungkin tidak mencerminkan kebutuhan sebenarnya.

b. Kompleksitas untuk Jumlah Tugas Besar: Untuk skenario dengan jumlah tugas yang sangat besar, kompleksitas ruang dan waktu dapat menjadi kendala. Pendekatan heuristik mungkin diperlukan untuk mengatasi skenario tersebut.

6. KESIMPULAN

Implementasi algoritma program dinamis untuk penjadwalan tugas harian membuktikan efektivitasnya dalam mengoptimalkan penggunaan waktu dengan memprioritaskan tugas-tugas yang paling penting. Berdasarkan analisis di atas, dapat disimpulkan beberapa poin penting:

6.1. Optimalitas dan Efisiensi:

Algoritma dinamis memberikan solusi optimal dalam batasan waktu yang ditentukan,

memastikan bahwa tugas dengan prioritas tertinggi diselesaikan terlebih dahulu. Kompleksitas waktu O(nW) masih efisien untuk jumlah tugas dan durasi yang wajar dalam konteks penjadwalan harian.

6.2. Kemudahan Implementasi:

Implementasi dalam bahasa Java menunjukkan bahwa algoritma ini tidak hanya teoretis tetapi juga praktis dan dapat diterapkan dalam kehidupan sehari-hari. Hal ini memberikan pengguna alat yang kuat untuk meningkatkan produktivitas mereka.

6.3. Adaptabilitas:

Algoritma ini dapat dengan mudah disesuaikan untuk skenario penjadwalan lainnya, seperti penjadwalan proyek, penjadwalan mingguan, atau bahkan penjadwalan dalam skala lebih besar. Fleksibilitas ini membuatnya sangat berguna dalam berbagai konteks manajemen waktu.

6.4. Keterbatasan yang Perlu Diatasi:

Beberapa keterbatasan, seperti subjektivitas dalam penentuan prioritas dan potensi masalah memori untuk jumlah tugas yang sangat besar, perlu dipertimbangkan. Pengguna dapat mengatasi ini dengan penilaian prioritas yang lebih objektif atau menggunakan pendekatan heuristik untuk skenario skala besar.

7. Rekomendasi untuk Pengembangan Lebih Lanjut:

Untuk meningkatkan kemampuan algoritma, integrasi dengan teknik lain seperti algoritma heuristik atau pendekatan machine learning dapat dieksplorasi. Ini dapat membantu mengatasi keterbatasan dalam kompleksitas ruang dan waktu serta penentuan prioritas yang lebih objektif.

Secara keseluruhan, algoritma program dinamis memberikan solusi yang efektif dan efisien untuk masalah penjadwalan tugas harian, membantu pengguna untuk mengelola waktu mereka dengan lebih baik dan meningkatkan produktivitas mereka secara keseluruhan.

8. UCAPAN TERIMA KASIH

Dengan rendah hati, penulis ingin menyampaikan terima kasih kepada berbagai pihak yang telah berkontribusi dalam penyelesaian makalah ini. Keberhasilan penulisan ini tidak hanya merupakan hasil usaha individu penulis, tetapi juga dukungan dan bimbingan dari beberapa pihak lain. Pertama-tama, penulis ingin bersyukur kepada Tuhan Yang Maha Esa atas limpahan rahmat dan rezekinya yang senantiasa mengiringi penulis dalam perjalanan penulisan makalah ini. Tidak lupa, penulis mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi, M.T. yang telah memberikan pengajaran dalam perkuliahan IF2211 Strategi Algoritma. Bimbingan dan pemahaman yang diberikan telah membantu penulis dalam menyusun makalah ini. Serta atas penyediaan sarana websitenya yang sangat bermanfaat selama proses pembelajaran. Sarana tersebut telah menjadi sumber referensi yang berharga bagi penulis. Terima kasih juga disampaikan kepada Nasywaa Anggun Athiefah, atas dukungannya dalam membantu penulis mengerjakan makalah ini. Harapan penulis, makalah ini dapat memberikan manfaat dan kontribusi positif. Terima kasih.

9. REFERENCES

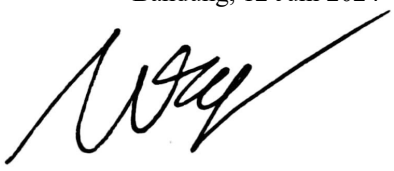
Munir, R. (n.d.). Program Dinamis (Dynamic Programming). informatika.stei.itb.ac.id. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Program-Dinamis-2020-Bagian2.pdf

Program Dinamis (Dynamic Programming) Bagian 1. (n.d.). Informatika. Retrieved June 12, 2024, from https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Program-Dinamis-2020-Bagian1.pdf

10. PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 12 Juni 2024



Mohammad Nugraha Eka Prawira 13522001