Algoritma Brute Force

Algoritma [brute force adalah](https://www.niagahoster.co.id/blog/brute-force-adalah/) jenis algoritma yang mencoba semua kemungkinan untuk menyelesaikan masalah. Dengan kata lain, ia akan melakukan percobaan satu persatu hingga menemukan solusi yang tepat.

Contoh penggunaan algoritma Brute Force misalnya untuk memecahkan 4 digit kode PIN dari angka 0 hingga 9. Setidaknya, butuh hingga 10.000 percobaan untuk menemukan kombinasi angka yang tepat.

Algoritma *brute force* bekerja dengan langkah-langkah sebagai berikut:

* **Inisialisasi**: Langkah pertama adalah inisialisasi, di mana semua parameter dan variabel yang diperlukan diatur dengan nilai awal.
* **Iterasi**: Algoritma akan memulai iterasi melalui semua kemungkinan secara berurutan.
* **Pengujian**: Pada setiap iterasi, algoritma akan menguji apakah solusi yang dihasilkan benar atau tidak.
* **Pemutakhiran**: Jika solusi yang ditemukan adalah yang benar, maka algoritma akan menghentikan iterasi dan menghasilkan hasil. Jika tidak, algoritma akan melanjutkan ke iterasi berikutnya.
* **Penyelesaian**: Algoritma akan menghasilkan solusi yang benar setelah mengecek semua kemungkinan.

### ****Contoh Penerapan Algoritma Brute Force****

#### ****Pencarian Kata dalam Teks****

Algoritma brute force dapat digunakan untuk mencari sebuah kata dalam teks. Algoritma ini akan mencoba semua kemungkinan letak kata dalam teks hingga menemukan kecocokan.

#### ****Pengujian Password****

Dalam keamanan komputer, algoritma brute force digunakan untuk menguji semua kombinasi kata sandi mungkin untuk mencoba membobol akun yang dilindungi kata sandi. Hal ini sering digunakan dalam uji penetrasi dan pemulihan kata sandi.

#### ****Pengoptimalan Perjalanan****

Algoritma brute force dapat digunakan untuk mengoptimalkan rute perjalanan dengan mencoba semua kemungkinan perjalanan dan memilih yang paling efisien.

Algoritma brute force memiliki beberapa kelebihan seperti:

* ***Kepastian Solusi***

***Salah satu keunggulan utama dari algoritma brute force adalah kepastian solusi. Algoritma ini akan selalu menemukan solusi jika ada solusi yang tersedia. Ini membuatnya sangat andal dalam situasi di mana kesalahan tidak dapat diterima.***

* *Kesederhanaan*

*Algoritma brute force adalah pendekatan yang mudah dimengerti dan diimplementasikan. Ini tidak memerlukan pengetahuan khusus atau teknik optimasi yang rumit.*

* *Universalitas*

*Algoritma brute force dapat diterapkan pada berbagai jenis masalah tanpa perlu modifikasi yang signifikan. Hal ini membuatnya serbaguna dalam berbagai konteks.*

Algoritma brute force  juga memiliki kekurangan seperti:

* *Kecepatan Rendah*

*Algoritma brute force cenderung lambat. Ini terutama terlihat pada masalah dengan ruang pencarian yang besar atau kombinatorial. Pencarian melalui semua kemungkinan secara berurutan memerlukan waktu yang signifikan.*

* *Konsumsi Sumber Daya Tinggi*

*Karena mencoba semua kemungkinan, algoritma brute force memerlukan banyak sumber daya komputasi. Hal ini dapat mengakibatkan penggunaan daya tinggi dan memerlukan perangkat keras yang kuat.*

* *Tidak Efisien untuk Masalah Kompleks*

*Untuk masalah dengan tingkat kompleksitas tinggi, algoritma brute force seringkali tidak efisien. Penyelesaian masalah dalam waktu yang wajar mungkin menjadi tidak mungkin dengan pendekatan ini.*

Dalam dunia kriptografi, algoritma brute force digunakan untuk mencoba semua kemungkinan kombinasi kata sandi hingga menemukan yang benar. Ini dapat digunakan untuk membuka akses ilegal ke sistem atau data yang dilindungi kata sandi. Meskipun metode ini lambat, itu bisa berhasil jika kata sandi relatif lemah. Selain itu, algoritma ini juga diterapkan dalam pencarian string dan pemrosesan teks, seperti mencari pola atau kata kunci dalam teks.

Algoritma brute force adalah metode sederhana yang dapat diterapkan dalam berbagai bidang untuk mencari solusi dengan mencoba semua kemungkinan. Namun, kegunaannya terbatas pada masalah yang relatif sederhana dan tidak efisien untuk masalah yang kompleks. Oleh karena itu, sebaiknya digunakan dalam situasi yang memerlukan kepastian solusi dan tidak memungkinkan penggunaan algoritma lain yang lebih efisien.

Sumber : [*https://www.perplexity.ai/search/algoritma-brute-force-algoritm-DZsA6m3wTy6aTBeXiUXk7A*](https://www.perplexity.ai/search/algoritma-brute-force-algoritm-DZsA6m3wTy6aTBeXiUXk7A)

[*https://www.niagahoster.co.id/blog/algoritma-pemrograman/*](https://www.niagahoster.co.id/blog/algoritma-pemrograman/)

Algoritma Recursive

Algoritme Recursive adalah  algoritma pemrograman yang dimana, ia menyelesaikan masalah dengan sedikit demi sedikit dengan cara membaginya ke dalam beberapa kondisi yang serupa.

Jenis-Jenis Algoritma Recursive

1. Algoritma Divide and Conquer

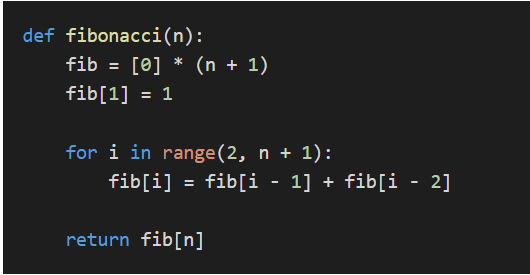
Membagi masalah menjadi dua atau lebih sub-masalah yang serupa dan lebih kecil, kemudian menyelesaikan dan menggabungkan solusi

Sebuah gambar berisi diagram, garis, Font, Rencana

Deskripsi dibuat secara otomatisContoh: Divide-and-conquer approach untuk mengurutkan daftar (38, 27, 43, 3, 9, 82, 10) dalam urutan menaik. *Bagian atas:* membagi menjadi subdaftar; *bagian tengah:* daftar satu elemen diurutkan dengan mudah; *bagian bawah:* menyusun subdaftar yang diurutkan.

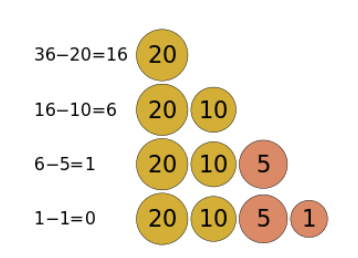
1. Algoritma Dinamis

Menggunakan teknik memoisasi, yaitu menyimpan hasil pemecahan masalah ke memori untuk selanjutnya digunakan lagi di masa mendatang.

Contoh: Pada contoh di ini, kita menggunakan pendekatan *bottom-up* untuk menghitung nilai Fibonacci ke-n. Dengan menyimpan hasil perhitungan sebelumnya, kita menghindari perhitungan berulang yang tidak perlu.

1. Algoritma Greedy

Bertolak belakang dengan algoritma dinamis, algoritma greedy justru tidak akan mempertimbangkan hasil pemecahan masalah sebelumnya untuk mengambil keputusan.

Contoh: Salah satu contoh penerapan algoritma greedy adalah permasalahan meminimalkan jumlah uang recehan logam sebagai hasil penukaran terhadap uang kertas seperti ditunjukan oleh Gambar 3 Jika tersedia uang pecahan logam senilai 1, 5, 10 dan 20, maka uang kertas senilai 36 dapat ditukar dengan cara berikut: Penetapan strategi greedy yaitu: pada setiap langkah, pilih koin dengan nilai terbesar dari himpunan koin yang tersisa, sehingga langkah yang diambil adalah sebagai berikut: 1. Langkah 1: pilih 1 buah koin bernilai 20 (Total = 20) 2. Langkah 2: pilih 1 buah koin bernilai 10 (Total = 20 + 10 = 30) 3. Langkah 3: pilih 1 buah koin bernilai 5 (Total = 20 + 10 + 5 = 35) 4. Langkah 4: pilih 1 buah koin bernilai 1 (Total = 20 + 10 + 5 + 1 = 36)

1. Algoritma Backtracking

Menyelesaikan masalah secara bertahap sambil mengeliminasi solusi yang ternyata tidak memecahkan masalah tersebut.

Sebuah gambar berisi teks, cuplikan layar, Font, Persegi

Deskripsi dibuat secara otomatisContoh: implementasi

Di sini, fungsi solve\_n\_queens\_util mencoba menempatkan ratu di setiap kolom dan melakukan backtrack jika penempatan tersebut tidak valid.

Kelebihan dan Kekurangan Algoritma Recursive:

Kelebihan:

1. Mengurangi kompleksitas waktu
2. Memberikan kejelasan lebih jelas daripada algoritma panjang karena biasanya function recursive relatif pendek secara code.
3. Membuat code lebih 'cantik'

Kekurangan:

1. Memakan lebih banyak memori
2. Justru membuat proses menjadi lebih lama apabila algoritma recursive nya tidak efisien

Dengan demikian, algoritma recursive, terutama algoritma Divide and Conquer, merupakan teknik yang powerful dalam pemrograman komputer untuk memecahkan masalah yang kompleks dengan cara membaginya menjadi sub-masalah yang lebih sederhana.

Sumber:

[*https://www.codepolitan.com/forum/thread/kelebihan-recursive-Sn1D/*](https://www.codepolitan.com/forum/thread/kelebihan-recursive-Sn1D/)

[*https://www.niagahoster.co.id/blog/algoritma-pemrograman/*](https://www.niagahoster.co.id/blog/algoritma-pemrograman/)

[*https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Algoritma-Runut-balik-(2018).pdf*](https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Algoritma-Runut-balik-(2018).pdf)

[*https://media.neliti.com/media/publications/280992-penerapan-algoritma-greedy-pada-penjadwa-386a4d99.pdf*](https://media.neliti.com/media/publications/280992-penerapan-algoritma-greedy-pada-penjadwa-386a4d99.pdf)

[*https://www.codepolitan.com/blog/apa-itu-dynamic-programmingpengertian-contoh-problem-dan-solusi/*](https://www.codepolitan.com/blog/apa-itu-dynamic-programmingpengertian-contoh-problem-dan-solusi/)

[*https://en.wikipedia.org/wiki/Divide-and-conquer\_algorithm*](https://en.wikipedia.org/wiki/Divide-and-conquer_algorithm)

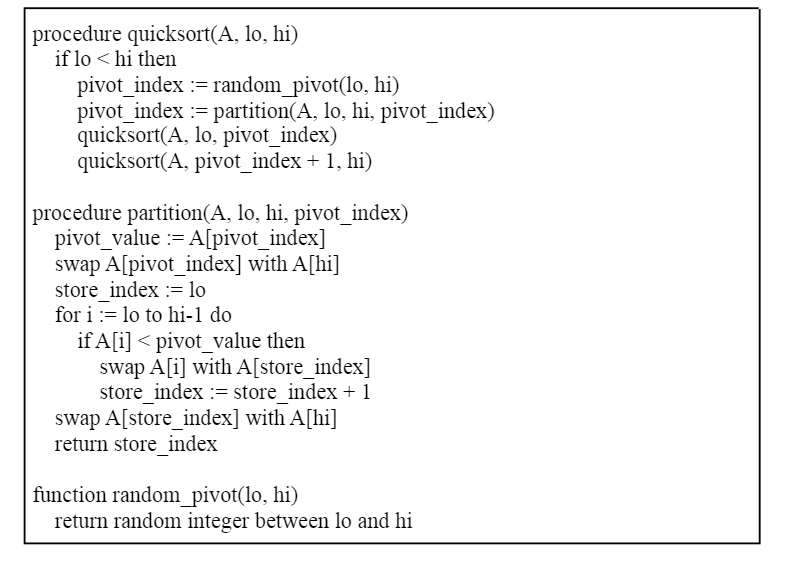
**Algoritma Randomized**

Algoritma randomized adalah logika pemrograman yang memanfaatkan nomor random guna menentukan langkah yang akan diambil selanjutnya. Metode pemecahan seperti ini bertujuan untuk meminimalisir kerumitan pemrograman. Contohnya, pemanfaatan nomor random yang muncul untuk menentukan pivot.

Fungsi Algoritma Randomized

* Fungsi utama algoritma randomized adalah untuk meningkatkan efisiensi dan kemampuan menyelesaikan masalah kompleks dengan cara menghindari pola yang berulang.

Contoh: Contoh algoritma acak adalah algoritma quick sort acak, yang merupakan variasidari algoritma quick sort standar.○Ide dasar dibalik quick sort adalah mempartisi array menjadi dua subarray, satuberisi elemen yang lebih kecil dari elemen pivot, dan yang lainnya berisi elemenyang lebih besar atau sama dengan pivot.○Pivot dipilih secara acak dalam quicksort acak, daripada menggunakan strategitetap atau deterministik.○Berikut contoh pseudocode untuk algoritma quick sort acak:



* Dalam contoh ini, fungsi quicksort mengambil tiga argumen: array A, dan indeks

bawah dan atas dari porsi array yang akan diurutkan.

* pertama-tama memeriksa apakah bagian yang akan disortir memiliki lebih dari

satu elemen, dan jika demikian, ia memilih indeks pivot acak menggunakan

fungsi random pivot.

* kemudian memanggil fungsi partisi untuk mempartisi array di sekitar pivot, dan
* secara rekursif memanggil dirinya sendiri pada dua subarray yang dibentuk oleh

partisi tersebut.

* Fungsi partisi membutuhkan empat argumen: larik A, indeks bawah dan atas dari

bagian larik yang akan dipartisi, dan indeks pivot.

Itu menukar elemen pivot dengan elemen terakhir di bagian array yang akan

dipartisi, dan kemudian menggunakan loop untuk beralih melalui elemen array.

* Jika sebuah elemen kurang dari nilai pivot, itu ditukar dengan elemen pada indeks

toko saat ini, dan indeks toko bertambah.

* Hasil akhirnya adalah elemen yang kurang dari pivot berada di sebelah kiri pivot

dan lebih besar dari pivot berada di sebelah kanan.

* Fungsi random\_pivot mengambil dua argumen: indeks bawah dan atas dari bagian

array yang akan diurutkan, dan mengembalikan bilangan bulat acak di antara

indeks tersebut, yang digunakan sebagai indeks pivot.

Kelebihan Dan kekurangan Randomized

* Algoritma acak memiliki beberapa kekuatan dan kelemahan yang harus

dipertimbangkan ketika memutuskan apakah mereka adalah pilihan yang tepat

untuk masalah tertentu.

Kekuatan:

* Mereka bisa lebih efisien dari Algoritma deterministik dalam kasus

tertentu. Misalnya, sortir cepat acak bisa lebih cepat dari sortir cepat

deterministik rata-rata, karena ini menghindari skenario terburuk di mana

pivot dipilih dengan buruk.

* Mereka bisa lebih kuat terhadap masukan musuh. Misalnya, Algoritma acak

tidak terlihat tidak Terpengaruh oleh penyerang jahat yang mencoba

membuatnya gagal.

* Mereka dapat berguna untuk memecahkan masalah yang belum diketahui

Algoritma deterministik efisiennya, seperti masalah memeriksa keutamaan

banyak sekali.

Kelemahan:

* Mereka dapat menghasilkan keluaran yang berbeda untuk Masukan yang

sama, yang dapat menjadi masalah jika Algoritma digunakan dalam

konteks di mana reproduktifitas penting.

* Mungkin Mereka tidak menemukan solusi optimal.
* Mereka bisa lebih kompleks untuk dirancang dan diimplementasikan

daripada algoritma deterministik.

* Mereka mungkin membutuhkan lebih banyak pengulangan untuk menemukan

Dengan demikian, algoritma randomized merupakan salah satu metode yang efektif dalam menyelesaikan masalah kompleks dengan menggunakan keputusan acak, seperti pada quick sort .

Sumber :

[*https://www.dewaweb.com/blog/mengenal-algoritma-pemrograman/*](https://www.dewaweb.com/blog/mengenal-algoritma-pemrograman/)

[*https://cmlabs.co/id-id/seo-terms/algoritma-dan-pemrograman*](https://cmlabs.co/id-id/seo-terms/algoritma-dan-pemrograman)

[*https://www.studocu.com/id/document/universitas-ahmad-dahlan/komputer-terapan/randomized/45023401*](https://www.studocu.com/id/document/universitas-ahmad-dahlan/komputer-terapan/randomized/45023401)

**Algoritma Sorting**

**Algoritma sorting adalah suatu metode atau teknik untuk mengurutkan data atau elemen-elemen dalam suatu [struktur data](https://fikti.umsu.ac.id/pengertian-fungsitipe-dan-cara-memilih-struktur-data/) secara teratur. Algoritma sorting merupakan salah satu konsep penting dalam pemrograman, tujuannya untuk mengubah data yang tidak teratur menjadi urutan yang teratur, misalnya dari data yang tidak terurut menjadi data yang terurut menaik atau menurun.**

**Jenis Algoritma Sorting**

* **Bubble Sort**

**Bubble Sort adalah salah satu algoritma sorting yang paling sederhana. Algoritma ini bekerja dengan membandingkan elemen-elemen berpasangan dalam daftar dan menukar mereka jika diperlukan. Proses ini terus berlanjut hingga semua elemen berada dalam urutan yang benar.**

* **Quick Sort**

**Quick Sort adalah algoritma yang mengurutkan dengan lebih efisien. Ia membagi daftar menjadi dua bagian, lalu mengurutkan setiap bagian secara terpisah. Kemudian, hasilnya digabungkan untuk mendapatkan daftar yang terurut.**

* **Merge Sort**

**Merge Sort juga membagi daftar menjadi dua bagian, tapi dia mengurutkan masing-masing bagian secara terpisah dan kemudian menggabungkannya untuk mendapatkan daftar yang terurut. Algoritma ini dikenal karena efisiensinya dalam mengurutkan data yang besar.**

* **Selection Sort**

**Selection Sort bekerja dengan cara mencari elemen terkecil dalam daftar dan menukarnya dengan elemen pertama. Proses ini terus berlanjut untuk elemen-elemen berikutnya hingga seluruh daftar terurut.**

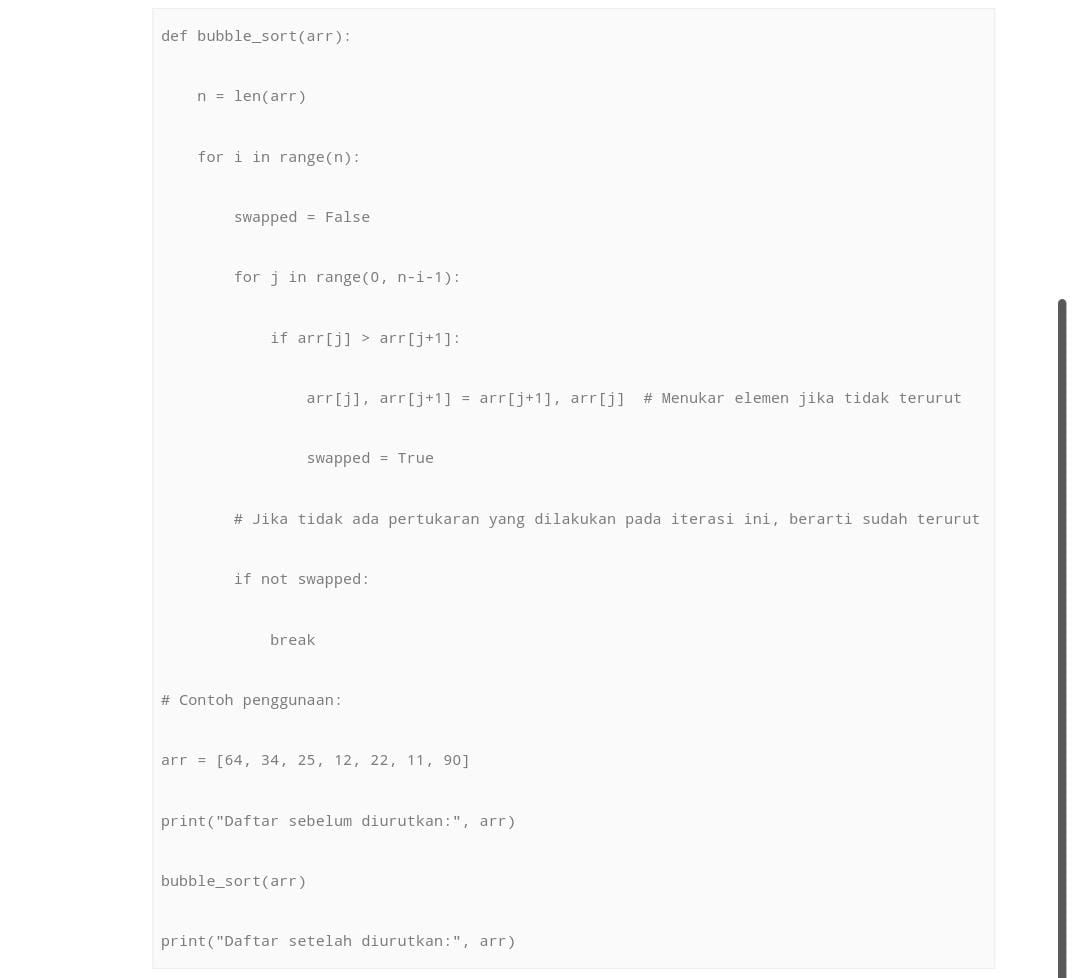
* **Insertion Sort**

**Insertion Sort membandingkan setiap elemen dengan elemen-elemen sebelumnya dan memasukkannya ke dalam posisi yang benar dalam daftar yang terurut. Algoritma ini efisien untuk daftar kecil.**

* **Shell Sort**

**Shell Sort adalah variasi dari Insertion Sort yang lebih efisien. Ia membandingkan dan menukar elemen-elemen yang terletak dalam jarak tertentu, lalu secara bertahap mengurutkan daftar tersebut hingga menjadi terurut.**

**Contoh program algoritma sorting**



**Fungsi dan Manfaat Sorting**

**Fungsi Pengurutan:**

1. **Mengorganisir Data: Pengurutan membantu mengorganisir data sehingga lebih mudah diakses dan dielola.**
2. **Memudahkan Pencarian: Data yang diurutkan memungkinkan untuk melakukan pencarian atau mencari elemen tertentu dengan lebih efisien, terutama jika data besar.**
3. **Mempermudah Analisis: Data yang terurut memungkinkan untuk menganalisis pola atau tren dengan lebih mudah dan cepat.**
4. **Meningkatkan Efisiensi Pemrosesan: Data yang terurut memungkinkan untuk menggunakan algoritma pencarian efisien seperti Binary Search, yang hanya dapat dilakukan pada data terurut.**

**Manfaat Pengurutan:**

1. **Optimasi Pencarian: Dengan data yang terurut, pencarian dapat dilakukan dengan cara yang lebih efisien, meminimalkan jumlah perbandingan yang diperlukan.**
2. **Mengoptimalkan Algoritma Lain: Banyak algoritma dan operasi lain memerlukan data yang terurut sebagai prasyarat. Misalnya, algoritma pencarian biner hanya dapat digunakan pada data terurut.**
3. **Peningkatan Kinerja Basis Data: Dalam basis data, pengurutan memungkinkan untuk mengakses dan memanipulasi data dengan lebih cepat, terutama dalam operasi seperti pengambilan data teratas (top-N queries).**
4. **Meningkatkan Efisiensi Proses Bisnis: Pengurutan dapat digunakan dalam berbagai konteks bisnis untuk mengoptimalkan proses, seperti mengurutkan faktur berdasarkan tanggal atau mengelompokkan pelanggan berdasarkan kriteria tertentu.**
5. **Mempermudah Presentasi Data: Dalam visualisasi data, pengurutan dapat membantu dalam menyajikan informasi dengan cara yang lebih terstruktur dan dapat dimengerti.**
6. **Membantu dalam Analisis Statistik: Dalam analisis data dan statistik, pengurutan dapat membantu dalam mengidentifikasi pola dan tren yang mungkin tidak terlihat dalam data yang tidak terurut.**
7. **Memfasilitasi Proses Pengolahan Data Lainnya: Data yang terurut dapat memberikan dasar yang lebih baik untuk berbagai operasi pengolahan data seperti penggabungan, perbandingan, atau pengelompokan.**

Kesimpulan

Algoritma pengurutan memainkan peranan penting dalam pengolahan data. Pemilihan algoritma yang tepat tergantung pada ukuran data dan kebutuhan spesifik aplikasi. Algoritma seperti Quick Sort dan Merge Sort sering digunakan untuk dataset besar karena efisiensinya dibandingkan dengan algoritma sederhana seperti Bubble Sort atau Selection Sort.

Sumber:

[*https://man2kulonprogo.sch.id/m2kp/2023/09/27/sorting/*](https://man2kulonprogo.sch.id/m2kp/2023/09/27/sorting/)

[*https://fikti.umsu.ac.id/pengertian-algoritma-sorting-pengurutan-dalam-pemrograman/*](https://fikti.umsu.ac.id/pengertian-algoritma-sorting-pengurutan-dalam-pemrograman/)

**Algoritma Searching**

Algoritma search adalah serangkaian langkah atau instruksi yang digunakan untuk mencari elemen atau informasi tertentu dalam suatu dataset. Tujuannya adalah untuk menemukan posisi atau keberadaan elemen yang dicari.

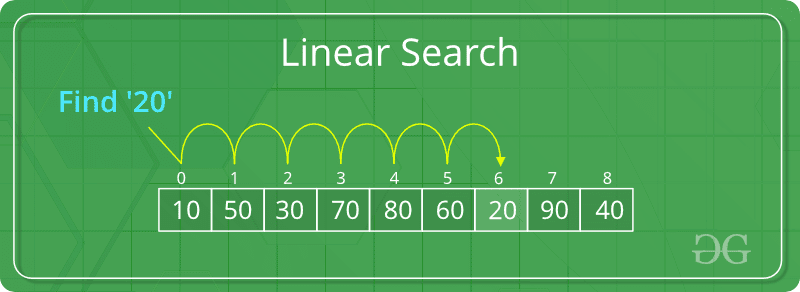
Dalam pemrograman, pencarian algoritma menjadi salah satu teknik penting dalam menyelesaikan berbagai masalah.Karakteristik Algoritma Pencarian

* Kompleksitas Waktu dan Ruang : Empat sifat penting dari algoritma pencarian adalah kelengkapan, optimalitas, kompleksitas waktu, dan kompleksitas ruang. Kompleksitas waktu Merujuk pada ukuran waktu bagi suatu algoritma untuk menyelesaikannya, sedangkan kompleksitas ruang Merujuk pada ruang penyimpanan maksimum yang diperlukan pada setiap titik selama pencarian.

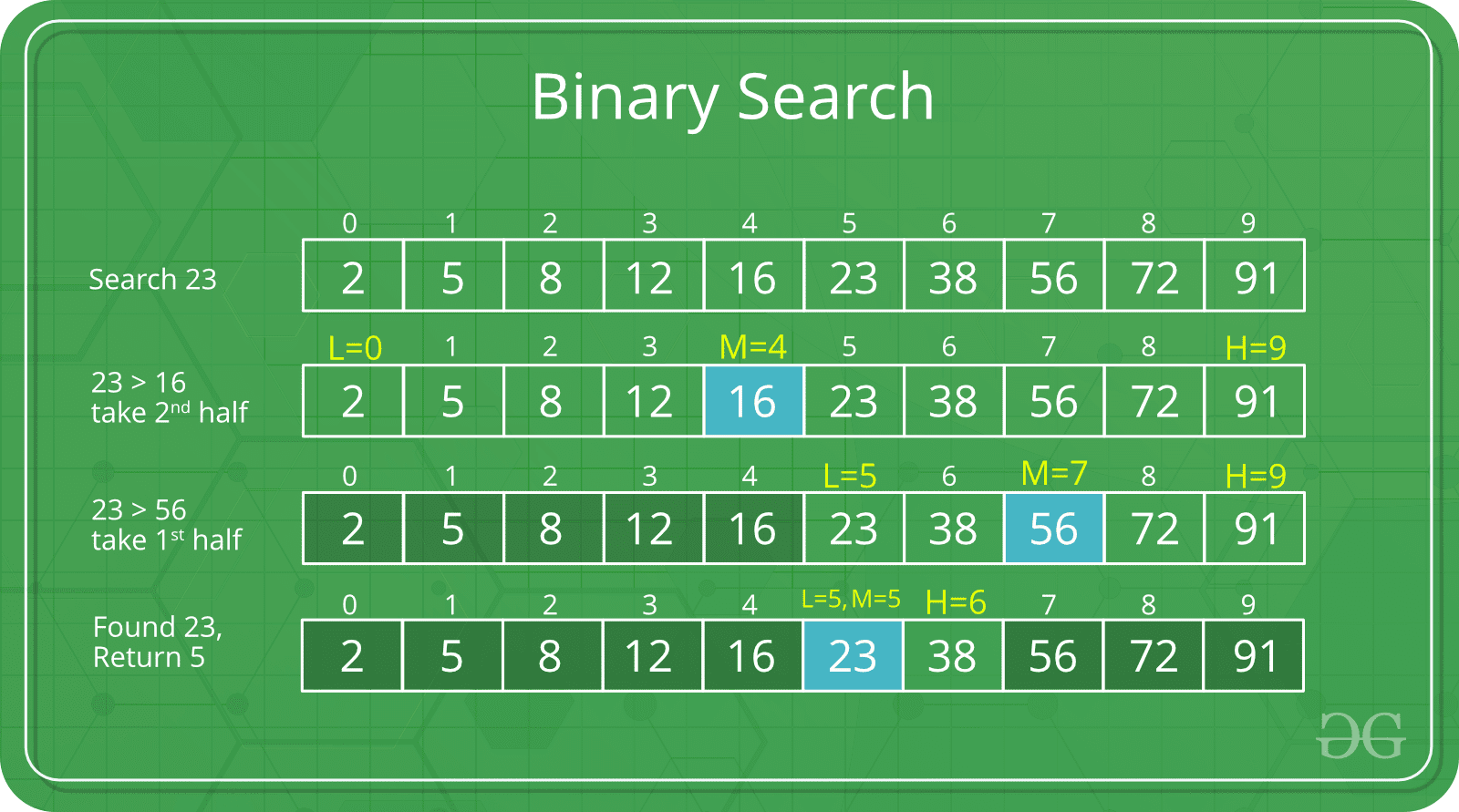
Jenis-jenis Algoritma Pencarian

Berdasarkan jenis operasi pencarian, algoritma ini umumnya diklasifikasikan menjadi dua jenis kategori:

1. **Sequential Search**: algoritma ini membandingkan setiap elemen array satu per satu secara berurutan, mulai dari elemen pertama, sampai elemen yang dicari ditemukan atau sampai semua elemen diperiksa. Jika elemen ditemukan, ia mengembalikan indeksnya, jika tidak -1. Contoh: Linear Search.



1. Interval Search: Algoritma ini digunakan untuk mencari elemen tertentu pada array/list dengan elemen yang berurutan yang mana elemen yang berada di dalam array harus diurutkan terlebih dahulu. Algoritma pencarian ini bekerja berdasarkan prinsip menemukan nilai tengah (M) pada array kemudian membagi menjadi dua bagian. Selanjutnya menentukan apakah nilai yang dicari lebih kecil (L) atau lebih besar (H) dari nilai yang ingin dicari. Apabila ditemukan kecocokan nilai maka akan mengembalikan output, jika tidak pencarian akan terus berlanjut hingga akhir dari pembagian jumlah elemen tersebut. Contoh: Binary Search.



Algoritma pencarian banyak digunakan dalam berbagai aplikasi, termasuk mesin pencari di internet, sistem informasi, dan aplikasi e-commerce. Efisiensi algoritma pencarian sangat mempengaruhi kecepatan dan kinerja sistem dalam mengakses dan mengolah data. Oleh karena itu, pemilihan algoritma pencarian yang tepat sangat penting untuk meningkatkan efisiensi dan efektivitas dalam pengolahan data.

Sumber:

[*https://fikti.umsu.ac.id/algoritma-search-pencarian-pengertian-jenis-dan-contoh-programmnya/*](https://fikti.umsu.ac.id/algoritma-search-pencarian-pengertian-jenis-dan-contoh-programmnya/)

<https://www.trivusi.web.id/2022/11/pengertian-algoritma-pencarian.html>

Algoritma Hashing

Algoritma hashing adalah suatu proses matematika yang mengubah data yang panjang menjadi sejumlah nilai tetap yang lebih pendek. Nilai tetap ini disebut hash atau message digest. Algoritma ini menggunakan fungsi hash untuk menghasilkan nilai hash yang unik untuk setiap input yang berbeda. Nilai hash yang dihasilkan bersifat deterministik, artinya input yang sama akan menghasilkan nilai hash yang sama pula.

**Jenis-jenis Algoritma Hashing**

Terdapat berbagai jenis algoritma ini yang umum digunakan, di antaranya adalah:

* **MD5 (Message Digest Algorithm 5)**

MD5 adalah salah satu algoritma hashing yang populer. Meskipun masih digunakan secara luas, MD5 diketahui memiliki kerentanan terhadap serangan collision dan sudah tidak dianggap aman untuk penggunaan yang kritis.

* **SHA-1 (Secure Hash Algorithm 1)**

SHA-1 adalah algoritma yang menghasilkan output hash sepanjang 160 bit. Meskipun SHA-1 lebih aman daripada MD5, namun algoritma ini juga telah diketahui memiliki kerentanan terhadap serangan collision.

* **SHA-256 (Secure Hash Algorithm 256-bit)**

SHA-256 adalah salah satu algoritma yang paling aman. Dengan ukuran hash sepanjang 256 bit, SHA-256 menghasilkan nilai hash yang sangat unik dan sulit untuk diretas.

* **Blake2**

Blake2 adalah algoritma modern yang dianggap lebih aman dan lebih cepat daripada SHA-256. Algoritma ini juga memiliki ukuran hash yang lebih fleksibel.

* **CRC (Cyclic Redundancy Check)**

CRC adalah algoritma yang umum digunakan dalam deteksi kesalahan dalam transmisi data. CRC menghasilkan nilai hash yang lebih pendek dibandingkan dengan algoritma hashing lainnya.

**Keuntungan Penggunaan Algoritma Hashing**

Penggunaan algoritma ini memiliki beberapa keuntungan, antara lain:

* Keamanan Data

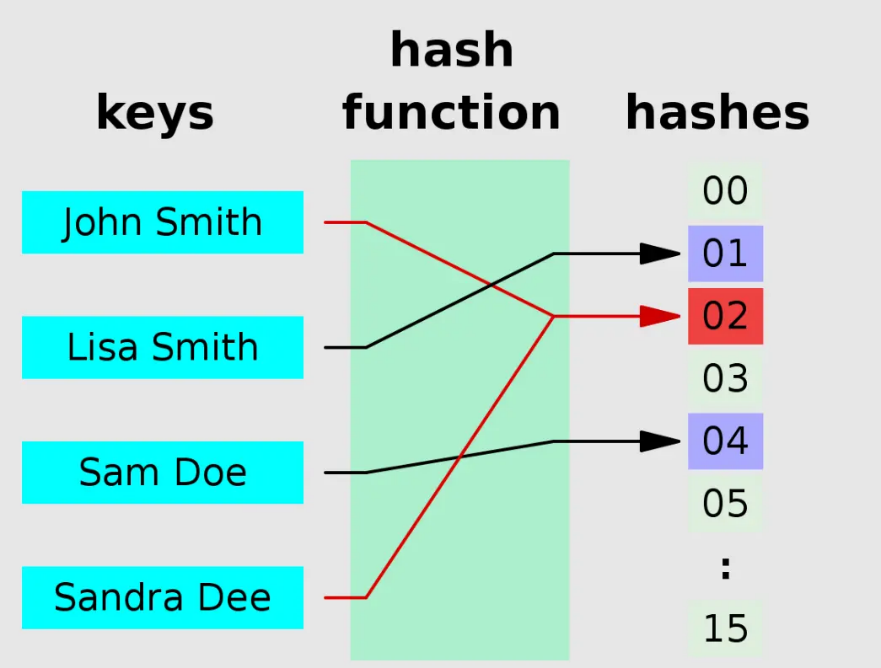
Algoritma hashing melindungi integritas data dengan menciptakan tanda tangan digital yang unik. Hal ini memungkinkan kita untuk mendeteksi adanya perubahan pada data dengan membandingkan hash yang dihasilkan sebelum dan setelah perubahan.

* Deteksi Perubahan Data

Ketika menggunakan algoritma hashing, jika terjadi perubahan sedikit pun pada data, hash yang dihasilkan akan berbeda secara signifikan. Hal ini memudahkan kita dalam mendeteksi manipulasi data yang tidak sah.

* Efisiensi Penyimpanan DataDalam beberapa kasus, penyimpanan hash yang relatif pendek bisa lebih efisien daripada penyimpanan data asli yang lebih panjang. Algoritma ini memungkinkan kita untuk menyimpan dan memperbandingkan hash dengan cepat, menghemat ruang penyimpanan.

Contoh pemanfaatan algoritma hashing yaitu untuk verifikasi password ketika ingin masuk ke akun tertentu.



**Kesimpulan**

Algoritma hashing adalah alat penting dalam dunia keamanan informasi. Dengan kemampuannya untuk menghasilkan nilai unik dari input data dan mendeteksi perubahan dengan cepat, algoritma ini memainkan peran krusial dalam berbagai aplikasi mulai dari verifikasi kata sandi hingga pengelolaan integritas file. Memilih algoritma hashing yang tepat sangat penting untuk memastikan keamanan dan efisiensi dalam pengolahan data.

Sumber:

[*https://kantinit.com/programming/apa-itu-algoritma-hashing-pengertian-jenis-dan-implementasi/*](https://kantinit.com/programming/apa-itu-algoritma-hashing-pengertian-jenis-dan-implementasi/)

[*https://www.niagahoster.co.id/blog/algoritma-pemrograman/*](https://www.niagahoster.co.id/blog/algoritma-pemrograman/)

**nama: hasanah, nim: D0424502, prodi: Sistem Informasi.**