Strategi Algoritma: Memecahkan Masalah

Algoritma adalah serangkaian langkah atau instruksi yang terstruktur dan logis untuk menyelesaikan sebuah masalah atau mencapai tujuan tertentu

kita akan belajar tentang dua strategi algoritma dasar yang sering dipakai: **Divide and Conquer** dan **Greedy Algorithm**. Jangan khawatir, kita akan bahas dengan bahasa yang mudah dimengerti!

---

1. Strategi "Divide and Conquer" (Bagi dan Kuasai)

Pernah dengar pepatah "Bagi dan Kuasai"? Konsep ini juga berlaku di dunia algoritma, lho! Strategi **Divide and Conquer** adalah cara memecahkan masalah besar dengan memecahnya menjadi masalah-masalah kecil yang lebih mudah diselesaikan. Setelah masalah kecil selesai, hasil dari masalah-masalah kecil itu digabungkan kembali untuk mendapatkan solusi masalah besar.

Contoh Gampangnya:

Bayangkan kamu punya setumpuk besar surat yang harus diurutkan berdasarkan abjad. Kalau kamu mengerjakannya sendirian, pasti butuh waktu lama dan pusing, kan?

Dengan strategi Divide and Conquer, kamu bisa:

1. **Bagi (Divide):** Bagi setumpuk surat itu menjadi beberapa tumpukan kecil yang bisa dikerjakan oleh beberapa temanmu.
2. **Kuasai (Conquer):** Masing-masing temanmu mengurutkan tumpukan kecil surat mereka sendiri. Ini jauh lebih mudah dan cepat!
3. **Gabung (Combine):** Setelah semua tumpukan kecil selesai diurutkan, gabungkan kembali semua tumpukan itu menjadi satu tumpukan besar yang sudah terurut rapi.

Kapan Strategi Ini Cocok Digunakan?

Strategi Divide and Conquer paling pas kalau masalahnya bisa dibagi menjadi bagian-bagian yang serupa dan bisa diselesaikan secara terpisah, lalu hasilnya digabungkan. Contoh lain yang sering pakai strategi ini adalah algoritma pengurutan data seperti **Merge Sort** atau **Quick Sort**.

---

2. Strategi "Greedy Algorithm" (Algoritma Serakah)

Nah, kalau yang satu ini namanya "Greedy" atau serakah. Jangan salah sangka dulu, serakah di sini maksudnya adalah mengambil keputusan terbaik yang terlihat saat ini, tanpa memikirkan konsekuensi jangka panjangnya. Mirip dengan prinsip "ambil yang paling menguntungkan sekarang juga!"

Contoh Gampangnya:

Misalkan kamu adalah kasir di toko, dan ada pelanggan yang membayar dengan uang Rp 10.000, tapi belanjaannya hanya Rp 3.500. Kamu harus mengembalikan Rp 6.500. Koin yang tersedia adalah Rp 5.000, Rp 1.000, Rp 500, Rp 200, dan Rp 100.

Dengan strategi Greedy Algorithm, kamu akan:

1. Ambil koin dengan nilai terbesar yang masih bisa digunakan: Rp 5.000 (sisa kembalian Rp 1.500).
2. Ambil lagi koin terbesar selanjutnya: Rp 1.000 (sisa kembalian Rp 500).
3. Ambil lagi koin terbesar selanjutnya: Rp 500 (sisa kembalian Rp 0).

Jadi, kembaliannya adalah satu koin Rp 5.000, satu koin Rp 1.000, dan satu koin Rp 500. Algoritma ini langsung memilih koin terbesar yang mungkin untuk mengurangi sisa kembalian.

Kapan Strategi Ini Cocok Digunakan?

Greedy Algorithm cocok untuk masalah di mana setiap langkah pengambilan keputusan lokal (saat itu juga) akan membawa kita pada solusi optimal secara keseluruhan. Namun, hati-hati! Tidak semua masalah bisa diselesaikan dengan Greedy Algorithm karena terkadang pilihan terbaik saat ini belum tentu jadi pilihan terbaik untuk keseluruhan masalah.

Mari Kita Coba Sendiri!

Sekarang giliran kalian untuk mencoba menerapkan strategi algoritma ini!

Soal Sederhana:

Kamu punya daftar angka: `[12, 5, 8, 20, 3, 15]`

Tugas: Urutkan angka-angka ini dari yang terkecil hingga terbesar menggunakan strategi **Divide and Conquer** atau strategi lain yang menurutmu cocok dan jelaskan langkah-langkahnya secara mandiri.

Tips:

- **Divide and Conquer:** Coba bagi daftar angka menjadi dua bagian, urutkan masing-masing bagian, lalu gabungkan.
- **Greedy Algorithm (jika kamu bisa memodifikasinya):** Kalau ini lebih sulit diterapkan langsung untuk pengurutan, tapi bisa dipakai untuk masalah lain seperti mencari nilai terbesar.

Selamat mencoba! Dengan memahami strategi algoritma ini