1. **Menentukan kompleksitas waktu dalam bahasa matlab**
2. Berikut ini adalah contoh algoritma dalam bahasa MATLAB untuk mengalikan dua matriks: Kompleksitas waktu algoritma perkalian matriks ini adalah O(mnp), di mana m, n, dan p adalah ukuran matriks (jumlah baris dan kolom). Pada kasus terburuk, ketika matriks A memiliki m baris, B memiliki n kolom, dan matriks B memiliki p kolom, kompleksitas waktu perkalian matriks adalah O(mnp).

### 2. Secara matematik

contoh algoritma yang cukup sederhana, yaitu penjumlahan semua elemen dalam sebuah array. Misalkan kita memiliki sebuah array dengan ukuran n.

Algoritma untuk menjumlahkan semua elemen dalam array tersebut adalah sebagai berikut:

Sekarang, mari kita hitung kompleksitas waktu algoritma ini secara matematik:

1. Langkah 1 adalah operasi sederhana dan tidak bergantung pada ukuran array. Jadi, kompleksitas waktu untuk langkah ini adalah   
   O(1).
2. Langkah 2 memerlukan iterasi melalui semua elemen dalam array, yaitu n elemen. Jadi, kompleksitas waktu untuk langkah ini adalah O  
   .

Karena langkah-langkah ini tidak saling bergantung satu sama lain (mereka tidak bersarang), kita hanya perlu mempertimbangkan kompleksitas waktu tertinggi, yaitu OJadi, secara matematik, kompleksitas waktu algoritma ini adalah O(1)+O=O. Ini berarti kompleksitas waktu algoritma penjumlahan array ini adalah linear terhadap ukuran array (n).

### ****Menggunakan Profiler****

MATLAB menyediakan alat Profiler yang dapat membantu Anda menganalisis penggunaan memori algoritma Anda. Anda dapat menjalankan Profiler untuk mengukur penggunaan memori selama eksekusi algoritma Anda.

Dengan menggunakan salah satu atau kombinasi dari metode di atas, Anda dapat memperoleh perkiraan kompleksitas ruang algoritma dalam bahasa MATLAB. Perlu diingat bahwa estimasi ini dapat bervariasi tergantung pada platform dan lingkungan eksekusi Anda.

1. Menghitung kompleksitas ruang secara matematik

Untuk menghitung kompleksitas ruang suatu algoritma secara matematik, Anda perlu mempertimbangkan jumlah memori yang digunakan oleh algoritma tersebut seiring dengan ukuran inputnya. Kompleksitas ruang biasanya diukur sebagai fungsi dari ukuran input.

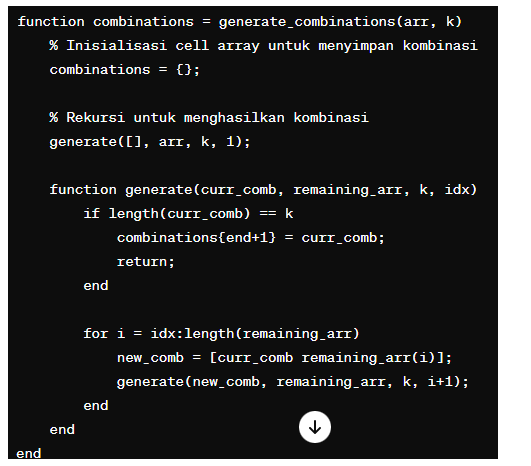
Berikut adalah langkah-langkah umum untuk menghitung kompleksitas ruang suatu algoritma secara matematik:

1. **Identifikasi Variabel dan Struktur Data**: Identifikasi semua variabel dan struktur data yang digunakan dalam algoritma Anda. Ini mencakup variabel sementara, array, matriks, pointer, dan struktur data lainnya.
2. **Estimasi Ukuran Memori**: Estimasi ukuran memori yang dibutuhkan untuk setiap variabel dan struktur data yang diidentifikasi pada langkah pertama. Misalnya, sebuah variabel integer biasanya menggunakan 4 byte memori, sebuah array dua dimensi mungkin menggunakan   
   ×size\_of\_elementn×m×size\_of\_element byte, dan sebagainya.
3. **Hitung Total Memori**: Hitung total memori yang digunakan oleh semua variabel dan struktur data dalam algoritma Anda.
4. **Tentukan Hubungan dengan Ukuran Input**: Tentukan hubungan antara ukuran input algoritma (misalnya, ukuran array, jumlah node dalam struktur data, dll.) dengan total memori yang digunakan. Ini akan memberi Anda kompleksitas ruang sebagai fungsi dari ukuran input.
5. **Sederhanakan dan Tentukan Notasi Asimtotik**: Sederhanakan ekspresi kompleksitas ruang dan tentukan notasi asimtotiknya. Ini memberi Anda pemahaman tentang bagaimana kompleksitas ruang algoritma Anda tumbuh seiring dengan ukuran inputnya.
6. **Contoh**: Misalkan Anda memiliki algoritma yang menggunakan matriksn×n dan tidak ada struktur data tambahan. Setiap elemen dalam matriks tersebut menggunakan 4 byte memori (asumsi bilangan floating point). Jumlah total memori yang digunakan oleh matriks tersebut adalah 4×4 x n byte. Dengan demikian, kompleksitas ruang algoritma ini adalah O(n2) byte.

Dengan mengikuti langkah-langkah ini, Anda dapat menghitung kompleksitas ruang algoritma Anda secara matematik. Perlu diingat bahwa kompleksitas ruang dapat bervariasi tergantung pada bagaimana algoritma Anda diimplementasikan dan bagaimana memori dialokasikan dan digunakan dalam lingkungan eksekusi tertentu.

### Mneghitung kompleksitas waktu dan ruang.

Mari kita ambil contoh sebuah algoritma, seperti algoritma untuk mencari semua kombinasi dari elemen-elemen dalam sebuah array dengan panjang tertentu. Kita akan menghitung kompleksitas waktu dan ruang untuk algoritma tersebut.



Sekarang, mari kita hitung kompleksitas waktu dan ruangnya:

**Kompleksitas Waktu:**

Untuk menghitung kompleksitas waktu, kita perlu memperhatikan seberapa sering blok kode di dalam fungsi generate dieksekusi. Dalam kasus ini, setiap panggilan rekursif pada fungsi generate memiliki kompleksitas waktu O no di mana n adalah panjang array remaining\_arr.

1. Panggilan pertama ke generate memiliki kompleksitas O no.
2. Setiap panggilan rekursif pada generate dilakukan dengan array yang lebih pendek, sehingga kompleksitas total rekursi adalah O(nk) di mana   
   k adalah panjang target kombinasi.
3. Jumlah total panggilan rekursif adalah sekitar   
   O(nk).

Jadi, kompleksitas waktu total untuk algoritma ini adalah O(nk).

**Kompleksitas Ruang:**

Untuk menghitung kompleksitas ruang, kita perlu memperhatikan berapa banyak memori yang digunakan untuk menyimpan kombinasi dan variabel-variabel lainnya.

1. Variabel combinations adalah cell array yang menyimpan semua kombinasi yang ditemukan. Jumlah maksimal kombinasi adalah C(n,k), di mana   
   n adalah panjang array input dan k adalah panjang target kombinasi.
2. Setiap kombinasi akan memiliki panjang   
   k, sehingga membutuhkan O(k) ruang.
3. Total kompleksitas ruang adalah  
   O(C(n,k)×k).Ini adalah contoh algoritma yang relatif rumit dalam hal kompleksitas waktu dan ruang. Dengan memperhatikan seberapa sering blok kode dieksekusi dan berapa banyak memori yang digunakan oleh variabel, Anda dapat menghitung kompleksitas waktu dan ruang untuk berbagai algoritma dalam MATLAB.