**STRUKTUR DATA**

**CONTOH BIG O NOTATION**



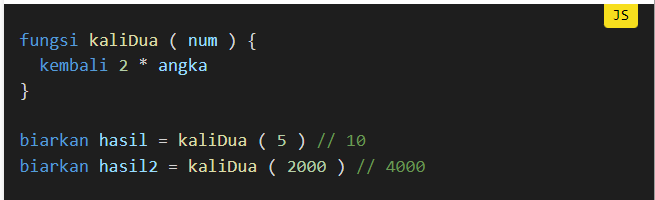
Nama : Safira Nur Rizqa Martalegawa

NPM : 22552011127

Mata Kuliah : Struktur Data

Kelas : TIF – RM 22A

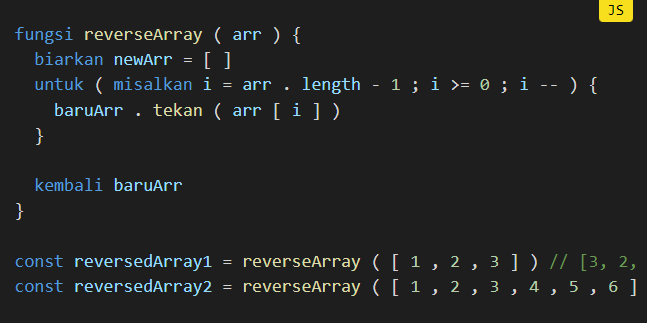
**1. O(1) — Constant Time**

****

Contoh diatas, Dibutuhkan bilangan bulat sebagai argumen, dan kemudian hanya mengembalikan 2 kali angka itu. Jadi, jika kita lulus dalam 5, maka akan kembali 2 kali 5, yaitu 10. Dan jika kita lulus pada tahun 2000, maka akan kembali 2 kali 2000, yaitu 4000.

1. **O(n) — Reverse Array**

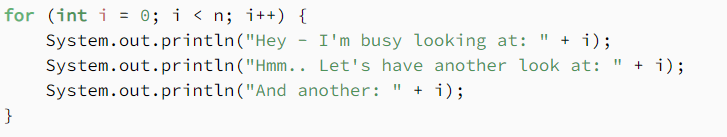
reverseArray, yang mengulang array input mulai dari item terakhir, dan membangun array baru yang akhirnya menjadi array input terbalik.



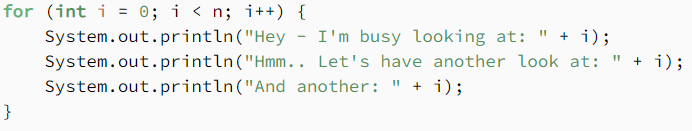
Jika kita memasukkan array dengan elemen 1, 2, 3, ia mengembalikan [3,2,1]. Dan jika kita masukan [1, 2 , 3 ,4 ,5, 6], itu kembali [6, 5, 4, 3, 2, 1].

**3. O(log n) — Linear Time**

Jika kita mengatakan sesuatu tumbuh secara linier, yang kita maksudkan adalah bahwa itu tumbuh berbanding lurus dengan ukuran inputnya.



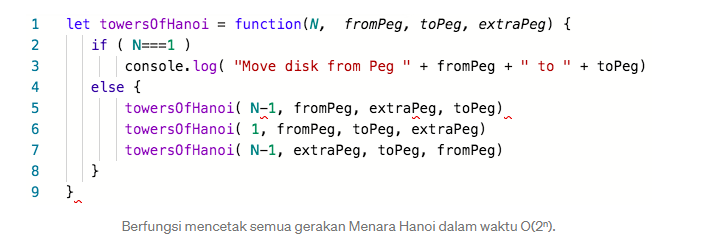
Kita ketahui bahwa algoritma sederhana yang disajikan di atas akan tumbuh secara linier dengan ukuran inputnya.



Runtime akan tetap linier dalam ukuran inputnya, n.

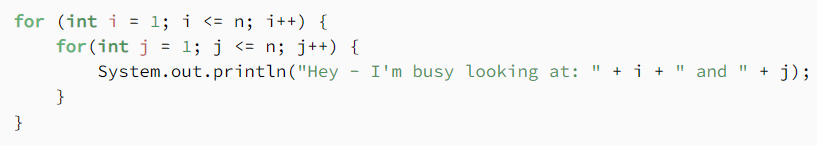
**4. O(Xⁿ)**

O(Xⁿ), dikenal sebagai waktu eksponensial , berarti waktu atau ruang akan dipangkatkan n. Waktu eksponensial sangat tidak efisien dan harus dihindari kecuali benar-benar diperlukan. Seringkali O(Xⁿ) dihasilkan dari memiliki algoritma rekursif yang memanggil sejumlah X algoritma dengan n-1. Towers of Hanoi adalah masalah terkenal yang memiliki solusi rekursif yang berjalan di O(2ⁿ).



## ****Polynomial Time Algorithms –****O(np)

Istilah polinomial adalah istilah umum yang mengandung fungsi kuadrat (n2), kubik (n3), kuartik (n4), dll. Yang penting untuk diketahui adalah bahwa O(n2) lebih cepat dari O(n3) yang lebih cepat dari O(n4), dll.

****

Algoritma ini akan berjalan 82= 64 kali. Catatan, jika kita membuat sarang untuk loop lain, ini akan menjadi algoritma O(n3).

## ****Exponential Time Algorithms –****O(****kn****)

Algoritma ini tumbuh secara proporsional dengan beberapa faktor yang dieksponenkan oleh ukuran input.

Misalnya, algoritma O(2n) berlipat ganda dengan setiap input tambahan. Jadi, jika n = 2, algoritma ini akan berjalan empat kali; jika n = 3, mereka akan berjalan delapan kali (seperti kebalikan dari algoritma waktu logaritmik).

Contoh sederhana dari algoritma waktu O(2n).

