ALGORITMA

Sequential dan Binary Search

Mata Kuliah Analisi Kompleksitas Algoritma

Binary Search

- Karena analisis dimulai dengan elemen tengah , dalam kasus terbaik, mungkin saja elemen tengah array itu sendiri adalah elemen target .
- Jika tidak, panjang array akan dibelah dua .
- Pada iterasi berikutnya, ukuran subarray diperkecil menggunakan hasil perbandingan sebelumnya.
    - Panjang awal array = N
    - Iterasi 1 - Panjang array = $n / 2$
    - Iterasi 2 - Panjang array = $(n/2) / 2 = n/2^2$
    - Iterasi k - Panjang array = $n / 2^k$

Setelah K iterasi, ukuran array menjadi 1 element

Panjang susunan = $N / 2^k$

⇨ $N = 2^k$

⇨ $\log_2(n) = \log_2 2^k$

⇨ $\log_2(n) = k * \log_2 2 = k$

⇨ $k = \log_2(n)$

```c
int binarySearch(int arr[], int size, int target) {
    int left = 0;
    int right = size - 1;

    while (left <= right) {
        int mid = left + (right - left) / 2;

        if (arr[mid] == target) {
            return mid; // Angka ditemukan, kembalikan indeks
        } else if (arr[mid] < target) {
            left = mid + 1; // Cari di setengah kanan
        } else {
            right = mid - 1; // Cari di setengah kiri
        }
    }

    return -1; // Angka tidak ditemukan
}
```

- kasus terbaik = *O(1)*
- kasus terburuk = *O (logn)*
- kasus rerata = *O (logn)*

```
ini binary
Masukkan angka yang ingin dicari: 2000
Angka 2000 ditemukan pada indeks 199

Process returned 0 (0x0)   execution time : 2.129 s
Press any key to continue.
```

Sequential Search

$$T(n) = \frac{(1 + 2 + 3 + \cdots + n)}{n}$$

$$= \frac{\frac{1}{2}n(1+n)}{n}$$

$$= \frac{(n+1)}{2} = (1)$$

$$= T(n) = O(n)$$

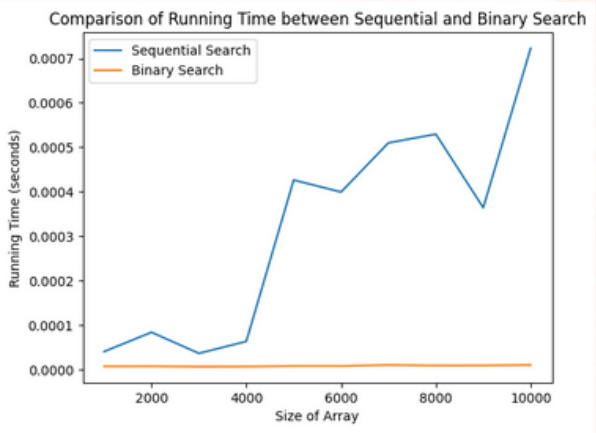
- kasus terbaik = *O(1)*
- kasus terburuk = *O(n)*
- kasus rerata = *O (n)*

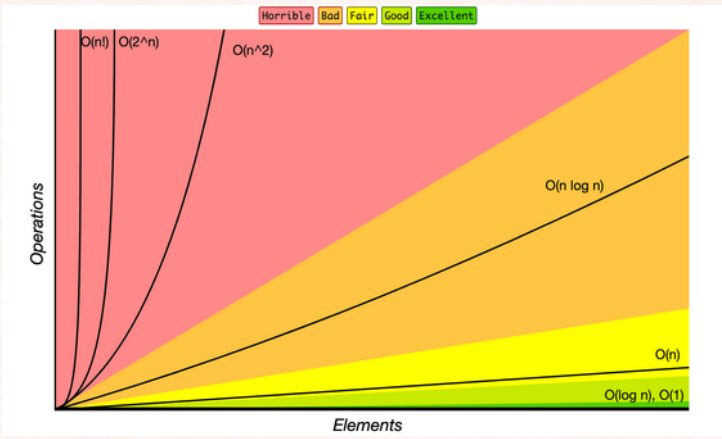
```c
int iterativeSequentialSearch(int arr[], int size, int key) {
    for (int i = 0; i < size; ++i) {
        if (arr[i] == key) {
            return i; // Angka ditemukan, mengembalikan indeks
        }
    }
    return -1; // Angka tidak ditemukan
}
```

```
ini sequential
Masukkan angka yang ingin dicari: 2000
Angka 2000 ditemukan pada indeks 199

Process returned 0 (0x0)   execution time : 2.720 s
Press any key to continue.
```

Grafik perbandingan





- Pada sequential search, waktu yang dibutuhkan untuk menyelesaikan pencarian meningkat secara linier seiring dengan peningkatan jumlah data yang dicari.
- Sedangkan pada binary search, waktu yang dibutuhkan meningkat secara logaritmik seiring dengan peningkatan jumlah data yang dicari.

pada chart Big O Complexity di atas karena binary search memiliki Big O *log(n)* dan sequential search memiliki Big O *O(n)* kami ambil kesimpulan bahwa Binary Search lebih cepat daripada Sequential Search karena Big O-nya tidak mendekati sumbu x