ALGORITMA
BINARY SEARCH

KELOMPOK 1

1. ABNER UMBU DONGGA P.
2. IFAT MARDIYAH
3. DHARMA PUTRANTO
4. MUHAMMAD ARIF SYAHRUDIN
5. RAFLI GILANG R.

BINARY SEARCH?

Binary Search adalah algoritma pencarian yang membagi 2 dari banyak elemen/data yang tersimpan di dalam array atau daftar yang sudah diurutkan.

Data yang ingin dicari **harus** terurut secara Ascending/Descending

CONTOH KASUS PENGGUNAAN *BINARY SEARCH*

Pencarian E-commerce: Fitur filter, memfilter produk "berdasarkan harga (termurah ke termahal)", Binary search dapat digunakan untuk menemukan rentang harga tertentu.

Mencari kata pada kamus digital: Pencarian kamus digital mengakses daftar kata "yang terurut secara alfabet (A-Z)", Binary search dapat digunakan untuk menemukan satu kata spesifik dengan cepat.

KELEBIHAN & KEKURANGAN

KELEBIHAN:

Efisiensi waktu: *Binary Search* memiliki kompleksitas waktu **O(Log n)** dimana **n** adalah jumlah dalam array. Hal ini membuatnya lebihi cepat dari pencarian **Linear Search (O (n))** terutama dalam dataset yang besar.

Skalabilitas: *Binary search* berskala baik dengan kumpulan data yang besar. Kinerjanya tetap konsisten meskipun data bertambah besar.

KEKURANGAN:

Data harus urut: *Binary Search* hanya bekerja Ketika data sudah terurut, jika tidak terurut algoritma akan gagal menghasilkan hasil yang diharapkan.

Tidak efisien untuk dataset kecil: *Binary search* tidak memberikan kecepatan yang signifikan ddibandingkan ***Linear Search***. Proses pembagian ruang pencarian secara berulang justru dapat menimbulkan overhead yang membuat pencarian menjadi kurang efisien.

CARA KERJA ?

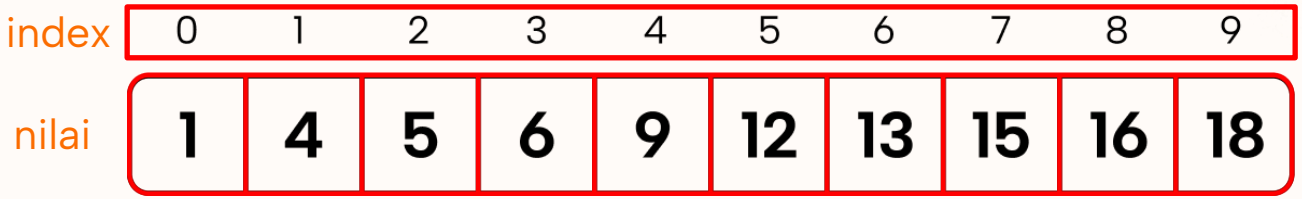
1. **Tentukan Rentang Pencarian Awal:** Awalnya, seluruh daftar dijadikan ruang pencarian.Rentang ini didefinisikan oleh dua indeks: **awal dan akhir,** yang menunjuk ke elemen pertama dan terakhir dalam daftar.

2. **Tentukan posisi tengah menggunakan rumus: tengah = (awal + akhir) / 2** Indeks ini menunjukkan elemen yang akan dibandingkan dengan target.

3. **Bandingkan *elemen* tengah dengan nilai yang dicari:**
   - Jika sama, elemen ditemukan, maka, pencarian selesai.
   - Jika lebih besar, lanjutkan pencarian ke bagian kiri daftar.
   - Jika lebih kecil, lanjutkan pencarian ke bagian kanan daftar.

4. **Ulangi Langkah 2–4:** Ulangi proses ini sampai ditemukan elemen yang dicari atau rentang pencarian menjadi kosong.

Target = 13

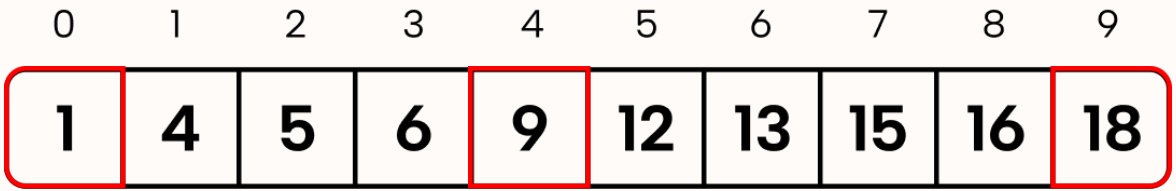


Target = 13

(awal + akhir) / 2 = tengah

(0 + 9) / 2 = 4,5

13 > 9

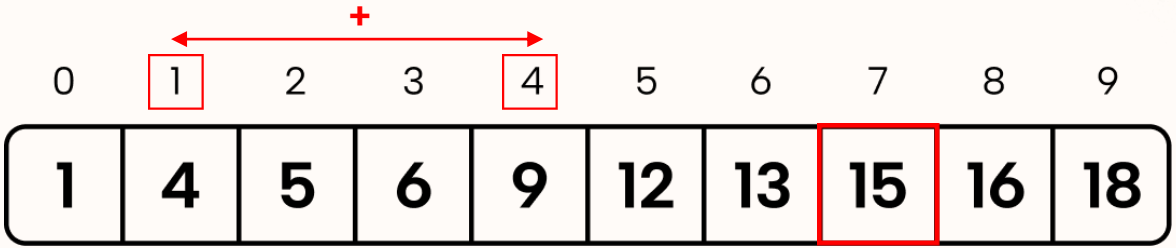


Target = 13

(awal + akhir) / 2 = tengah

(5 + 9) / 2 = 7

13 < 15

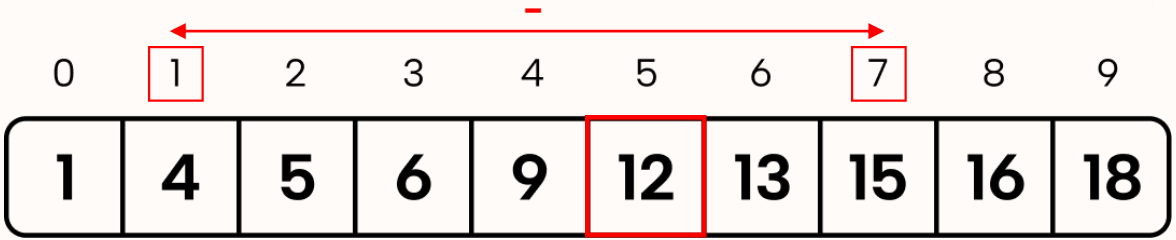


Target = 13

(awal + akhir) / 2 = tengah

(5 + 6) / 2 = 5,5

13 > 12

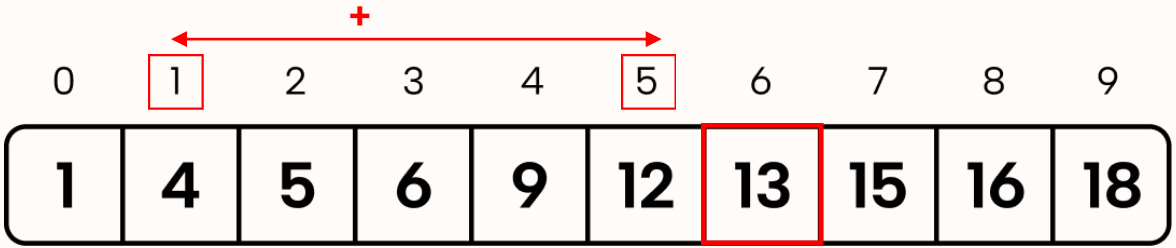


Target = 13

(awal + akhir) / 2 = tengah

(6 + 6) / 2 = 6

13 = 13



```cpp
#include <iostream>
#include <algorithm>
#include <string>
// #include <vector>

using namespace std;

// ascending - integer
int binarySearch(int arr[], int awal, int akhir, int target){
    while(awal <= akhir){
        int tengah = (awal + akhir) / 2;
        if(target == arr[tengah]){
            return tengah;
        } else if(target < arr[tengah]){
            akhir = tengah - 1;
        } else if(target > arr[tengah]){
            awal = tengah + 1;
        }
    }
    return -1;
}
```

```cpp
int main(){
    int arr[]= {1, 3, 6, 9, 8, 30, 27, 14, 25, 40, 10, 2, 60, 100, 90};
    int n = sizeof(arr) / sizeof(arr[0]);
    sort(arr, arr + n);

    int target;
    cout << "Masukan target elemen yang dicari: "; cin >> target;

    int result = binarySearch(arr, 0, n-1, target);


    for (int i = 0; i < n;i++){
        cout << arr[i] << " ";
    }
    if (result != -1) {
        cout << "\n\nNilai " << target << " ditemukan di indeks ke-" << result << endl;
    } else {
        cout << "\n\nNilai " << target  << " tidak ditemukan dalam array" << endl;
    }

    return 0;

}
```

KESIMPULAN

Binary Search adalah algoritma yang sangat berguna untuk memecahkan masalah dalam proses pencarian data yang sudah terurut, dengan efisiensi kompleksitas waktu yang sangat baik.

Algoritma ini bekerja dengan membagi dua bagian kumpulan data dan juga bekerja dengan sangat baik dalam dataset yang besar. Dibalik kelebihan algoritma ini juga memiliki kekurangan yaitu tidak coock digunakan dalam data kecil dan data harus berurut.

Algoritma ini umum dipakai dalam kehidupan nyata misal dalam pencarian item atau barang di aplikasi e-commerce.

THANKS