Ilham Kukuh Febriyansyah // 2320506043

Rangkuman Week 4

Algoritma pencarian dan pengurutan adalah bagian penting dari pemrograman komputer dan sering digunakan dalam pengembangan perangkat lunak. Berikut adalah rangkuman tentang algoritma pencarian dan pengurutan dalam bahasa pemrograman Python:

Algoritma Pencarian:

1. Linear Search (Pencarian Linear):

- Mencari elemen satu per satu secara berurutan.

- Cocok untuk daftar kecil atau tidak terurut.

- Contoh implementasi:

python

def linear\_search(arr, target):

for i in range(len(arr)):

if arr[i] == target:

return i

return -1

2. Binary Search (Pencarian Biner):

- Hanya dapat digunakan pada daftar terurut.

- Membandingkan elemen tengah dan memutuskan apakah mencari di setengah kiri atau kanan.

- Contoh implementasi:

python

def binary\_search(arr, target):

low, high = 0, len(arr) - 1

while low <= high:

mid = (low + high) // 2

if arr[mid] == target:

return mid

elif arr[mid] < target:

low = mid + 1

else:

high = mid - 1

return -1

Algoritma Pengurutan:

1. Bubble Sort (Pengurutan gelembung):

- Bandingkan dan tukar elemen berdekatan yang tidak sesuai.

- Iteratif hingga tidak ada pertukaran yang diperlukan.

- Contoh implementasi:

def bubble\_sort(arr):

n = len(arr)

for i in range(n - 1):

for j in range(0, n - i - 1):

if arr[j] > arr[j + 1]:

arr[j], arr[j + 1] = arr[j + 1], arr[j]

2. Selection Sort (Pengurutan Pilihan):

- Pilih elemen minimum dari sisa daftar dan tukar dengan elemen pertama.

- Iteratif hingga seluruh daftar terurut.

- Contoh implementasi:

def selection\_sort(arr):

n = len(arr)

for i in range(n):

min\_idx = i

for j in range(i+1, n):

if arr[j] < arr[min\_idx]:

min\_idx = j

arr[i], arr[min\_idx] = arr[min\_idx], arr[i]

3. Insertion Sort (Pengurutan Sisipan):

- Ambil elemen satu per satu dan sisipkan ke posisi yang sesuai di daftar terurut.

- Cocok untuk daftar yang hampir terurut.

- Contoh implementasi:

def insertion\_sort(arr):

for i in range(1, len(arr)):

key = arr[i]

j = i - 1

while j >= 0 and key < arr[j]:

arr[j + 1] = arr[j]

j -= 1

arr[j + 1] = key

Penting untuk memilih algoritma yang tepat tergantung pada karakteristik data yang dihadapi dan kebutuhan kinerja aplikasi.