Nama : Wita Adelia

Nim : 20220046

MataKuliah : Praktikum Desain dan Analisis Algoritma

LAPORAN ALGORITMA ANALISISA KASUS TERBURUK, RATA-RATA DAN TERBAIK

1. Konsep Dasar Analisis Kasus Terburuk, Rata-rata, dan Terbaik:
   a. Kasus Terburuk (Worst Case): Merupakan analisis yang dilakukan untuk mengetahui performa terburuk suatu algoritma pada situasi paling buruk, yaitu ketika input yang diberikan memiliki ukuran atau karakteristik yang paling tidak menguntungkan bagi algoritma tersebut.
   b. Kasus Rata-rata (Average Case): Merupakan analisis yang dilakukan untuk mengetahui performa algoritma pada input yang memiliki ukuran atau karakteristik secara acak atau dalam distribusi yang seragam.
   c. Kasus Terbaik (Best Case): Merupakan analisis yang dilakukan untuk mengetahui performa terbaik suatu algoritma pada situasi paling menguntungkan, yaitu ketika input yang diberikan memiliki ukuran atau karakteristik yang paling menguntungkan bagi algoritma tersebut.

2. Teknik Analisis Kasus Terburuk, Rata-rata, dan Terbaik:
   1) Kasus Terburuk (Worst Case): Teknik ini melibatkan identifikasi dan analisis dari operasi-operasi algoritma yang membutuhkan waktu paling lama untuk menyelesaikan tugas pada situasi input yang paling tidak menguntungkan. Biasanya, analisis dilakukan dengan mempertimbangkan ukuran input terbesar atau input yang memicu jumlah operasi paling banyak.
   2) Kasus Rata-rata (Average Case): Teknik ini melibatkan analisis statistik dengan mempertimbangkan distribusi input yang mungkin terjadi. Dalam beberapa kasus, teknik ini melibatkan perhitungan matematis menggunakan metode seperti ekspektasi.
   3) Kasus Terbaik (Best Case): Teknik ini melibatkan identifikasi dan analisis dari operasi-operasi algoritma yang membutuhkan waktu paling sedikit untuk menyelesaikan tugas pada situasi input yang paling menguntungkan. Biasanya, analisis dilakukan dengan

mempertimbangkan ukuran input terkecil atau input yang memicu jumlah operasi paling sedikit.

3. Latihan

   Kasus terburuk Kasus terburuk pada algoritma sorting adalah ketika elemen sudah terurut terbalik. Contoh implementasi kasus terburuk pada Bubble Sort, Merge Sort, dan Insertion Sort adalah sebagai berikut:

   - Bubble Sort:

```
def bubbleSort(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]
    return arr

# Contoh penggunaan fungsi bubbleSort
arr = [64, 34, 25, 12, 22, 11, 90]
sorted_arr = bubbleSort(arr)
print("Hasil Bubble Sort:", sorted_arr)
```

   Dengan hasil Input

```
Hasil Bubble Sort: [90, 64, 34, 25, 22, 12, 11]
```

   Pada kasus terburuk, elemen-elemen dalam larik diurutkan terbalik. Algoritma Bubble Sort akan membandingkan pasangan elemen sepanjang larik dan menukar posisi jika elemen sebelumnya lebih kecil dari elemen berikutnya. Dalam implementasi ini, jika elemen sebelumnya lebih kecil, maka pertukaran akan dilakukan. Namun, dalam kasus terburuk, algoritma akan melakukan pertukaran pada setiap langkah, sehingga jumlah langkah yang diperlukan akan maksimal.

- Merge Sort:

```
def mergeSort(arr):
    if len(arr) > 1:
    mid = len(arr)//2
    L = arr[:mid]
    R = arr[mid:]

    mergeSort(L)
    mergeSort(R)

    i = j = k = 0

    while i < len(L) and j < len(R):
        if L[i] > R[j]:
            arr[k] = L[i]
            i += 1
        else:
            arr[k] = R[j]
            j += 1
            k += 1

        while i < len(L):
            arr[k] = L[i]
            i += 1
            k += 1

        while j < len(R):
            arr[k] = R[j]
            j += 1
            k += 1
return arr
```

Menambahkan tanda kurung di sekitar pembagian mid = len(arr) // 2 untuk memastikan bahwa pembagian dilakukan dengan benar.

- Insertion Sort:

```
def insertionSort(arr):
    for i in range(1, len(arr)):
        key = arr[i]
        j = i - 1
        while j >= 0 and arr[j] < key:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key
    return arr
```

Pada kasus terburuk, elemen-elemen dalam larik diurutkan terbalik. Algoritma Insertion Sort membandingkan setiap elemen dengan elemen-elemen sebelumnya dan menyisipkannya ke posisi yang tepat. Dalam implementasi ini, elemen-elemen dibandingkan dengan elemen-elemen sebelumnya dalam larik secara berurutan,