NAMA : HIJRA S. OTJI

NIM : F55121051

KELAS : TI_B

Tugas PAA II

Membuat kode program menggunakan algoritma Bubble Sort dan Insertion Sort, dan membandingkan algoritma manakah yang lebih optimal.

1. Bubble Sort

- Kode Program

BubbleSort.py InsertionSort.py

```python
import time

def bubble_sort(arr):
    n = len(arr)

    start_time = time.time()  # Waktu mulai eksekusi

    for i in range(n - 1):
        for j in range(n - i - 1):
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]

                # Menampilkan proses iterasi
                print(arr)

    end_time = time.time()  # Waktu selesai eksekusi
    execution_time = end_time - start_time  # Menghitung waktu eksekusi

    return arr, execution_time


# Contoh penggunaan:
arr = [4, 1, 2, 3, 5]
sorted_arr, execution_time = bubble_sort(arr)
print("Hasil akhir:", sorted_arr)
print("Waktu eksekusi:", execution_time, "detik")
```

- Hasil Run

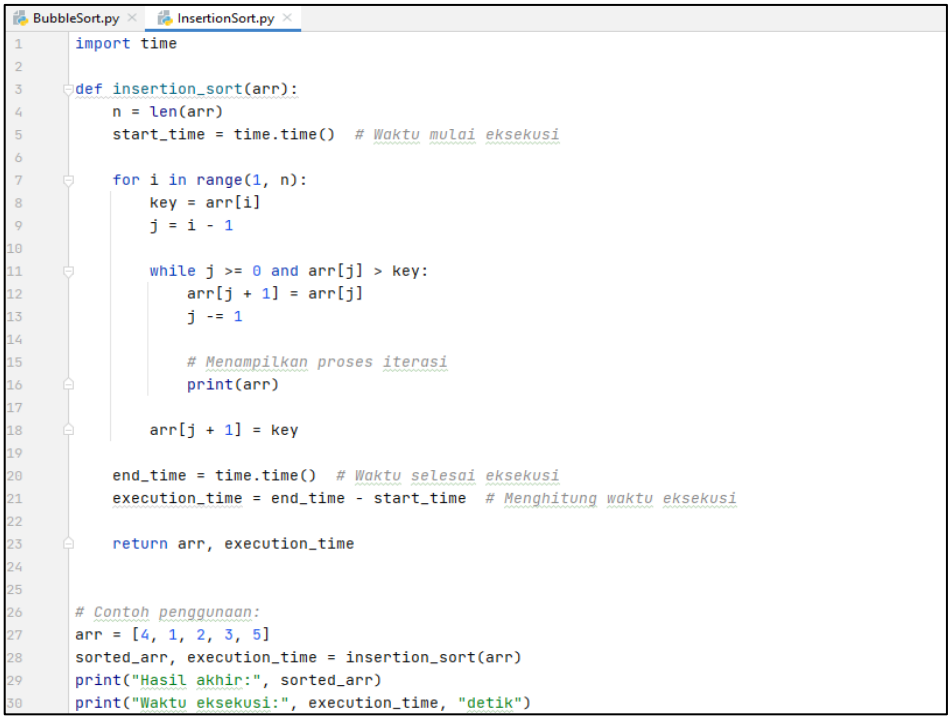
Run: BubbleSort

```
C:\Users\User\PycharmProjects\BubbleSort_Dan_InsertionSort\venv\Scripts\python.exe C:\Users\User\PycharmProjects\BubbleSort_Dan_InsertionSort\BubbleSort.py
[1, 4, 2, 3, 5]
[1, 2, 4, 3, 5]
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5]
[1, 2, 3, 4, 5]
Hasil akhir: [1, 2, 3, 4, 5]
Waktu eksekusi: 0.0 detik

Process finished with exit code 0
```

2. Insertion Sort

- Kode Program



```python
import time

def insertion_sort(arr):
    n = len(arr)
    start_time = time.time()  # Waktu mulai eksekusi

    for i in range(1, n):
        key = arr[i]
        j = i - 1

        while j >= 0 and arr[j] > key:
            arr[j + 1] = arr[j]
            j -= 1

            # Menampilkan proses iterasi
            print(arr)

        arr[j + 1] = key

    end_time = time.time()  # Waktu selesai eksekusi
    execution_time = end_time - start_time  # Menghitung waktu eksekusi

    return arr, execution_time


# Contoh penggunaan:
arr = [4, 1, 2, 3, 5]
sorted_arr, execution_time = insertion_sort(arr)
print("Hasil akhir:", sorted_arr)
print("Waktu eksekusi:", execution_time, "detik")
```

- Hasil Run

```
C:\Users\User\PycharmProjects\BubbleSort_Dan_InsertionSort\venv\Scripts\python.exe C:\Users\User\PycharmProjects\BubbleSort_Dan_InsertionSort\InsertionSort.py
[4, 4, 2, 3, 5]
[1, 4, 4, 3, 5]
[1, 2, 4, 4, 5]
Hasil akhir: [1, 2, 3, 4, 5]
Waktu eksekusi: 0.0 detik

Process finished with exit code 0
```

Penjelasan :

Dalam kasus ini, keduanya bukan termasuk kasus terbaik (best case) untuk masing-masing algoritma. Best case untuk Bubble Sort adalah ketika array sudah terurut secara membesar, sehingga tidak ada perubahan yang harus dilakukan pada iterasi pertama. Best case untuk Insertion Sort adalah ketika array juga sudah terurut secara membesar, di mana hanya perbandingan dan pergeseran minimal yang diperlukan.

Namun, jika kita melihat keduanya dalam kasus yang diberikan (array [4, 1, 2, 3, 5]), kita dapat mengamati perbedaan dalam kinerja mereka:

1. Insertion Sort: Algoritma ini membutuhkan lebih sedikit perbandingan dan pergeseran data pada setiap iterasi ketika mendekati keadaan terurut. Pada

kasus ini, setiap elemen akan dimasukkan ke posisi yang tepat secara berurutan. Jumlah iterasi yang dibutuhkan untuk mengurutkan array ini adalah 4, dan algoritma ini menunjukkan kinerja yang lebih baik dibandingkan dengan Bubble Sort dalam hal jumlah iterasi.

2. Bubble Sort: Algoritma ini melakukan perbandingan dan penukaran pasangan elemen yang berdekatan sampai seluruh array terurut. Dalam kasus ini, Bubble Sort akan melakukan 4 iterasi untuk mengurutkan array. Pada setiap iterasi, elemen terbesar akan naik ke posisi yang benar. Namun, dalam hal kinerja, Bubble Sort membutuhkan lebih banyak perbandingan dan pergeseran data dibandingkan dengan Insertion Sort.

Jadi, dalam kasus ini, algoritma yang lebih optimal adalah Insertion Sort karena kinerjanya yang lebih baik dalam hal jumlah perbandingan dan pergeseran data yang diperlukan untuk mengurutkan array tersebut.