Penjelasan algoritma dari metode Bubble sort, Selection Sort dan Insertion Sort

Praktikum 5

G.211.22.0002

Bubble Sort

Langkah-langkah:

1. Tentukan panjang daftar n.
2. Mulai iterasi dari indeks pertama hingga indeks n-1.
3. Dalam setiap iterasi pertama, atur variabel swapped menjadi False. Ini digunakan untuk memeriksa apakah ada pertukaran elemen dalam satu iterasi.
4. Mulai iterasi kedua di dalam iterasi pertama dari indeks 0 hingga n-i-1.
5. Bandingkan elemen ke-j dengan elemen ke-(j+1). Jika elemen ke-j lebih besar dari elemen ke-(j+1), tukar posisi keduanya dan atur swapped menjadi True.
6. Ulangi langkah 4-5 hingga seluruh daftar terurut satu kali.

Jika tidak ada pertukaran elemen dalam satu iterasi penuh, berarti daftar sudah terurut, dan dapat keluar dari loop.

```python
def bubbleSort(arr):
    n = len(arr)
    swapped = False

    for i in range(n-1):
        for j in range(0, n-i-1):
            if arr[j] > arr[j + 1]:
                swapped = True
                arr[j], arr[j + 1] = arr[j + 1], arr[j]

        if not swapped:
            break

# Driver code to test above
arr = [64, 34, 25, 12, 22, 11, 90]
bubbleSort(arr)
print("Sorted array is:")
for i in range(len(arr)):
    print("% d" % arr[i], end=" ")
```

```
Sorted array is:
 11  12  22  25  34  64  90
```

Pada akhirnya, setelah Bubble Sort selesai, elemen-elemen dalam array arr akan diurutkan

Selection Sort

Langkah-langkah:

1. Tentukan panjang daftar n.
2. Mulai iterasi dari indeks pertama hingga indeks n-1.
3. Di setiap iterasi pertama, atur variabel min_index ke indeks saat ini.
4. Mulai iterasi kedua di dalam iterasi pertama dari indeks (i+1) hingga n-1.
5. Bandingkan elemen di indeks j dengan elemen di min_index. Jika elemen di indeks j lebih kecil dari elemen di min_index, atur min_index ke j.
6. Setelah selesai iterasi kedua, tukar elemen di indeks i dengan elemen di min_index jika min_index bukanlah indeks awal.
7. Ulangi langkah 3-6 hingga seluruh daftar terurut satu kali.

```python
def selectionSort(arr):
    n = len(arr)

    for i in range(n):
        min_index = i

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

        arr[i], arr[min_index] = arr[min_index], arr[i]

# Driver code to test Selection Sort
arr = [64, 34, 25, 12, 22, 11, 90]
selectionSort(arr)
print("Sorted array is:")
for i in range(len(arr)):
    print("% d" % arr[i], end=" ")
```

```
Sorted array is:
 11  12  22  25  34  64  90
```

Insertion sort

Langkah-langkah:

Mulai iterasi dari indeks kedua (indeks 1) hingga indeks terakhir dalam daftar.

Pilih elemen saat ini dan simpan dalam variabel key.

Pilih indeks sebelum elemen saat ini (j = i - 1).

Selama j tidak kurang dari 0 dan elemen di indeks j lebih besar dari key, pindahkan elemen di indeks j ke indeks (j + 1).

Setelah langkah 4, dekremen j.

Tempatkan key pada indeks (j + 1).

Ulangi langkah 2-6 untuk setiap elemen dalam daftar.

```python
def insertionSort(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

# Driver code to test Insertion Sort
arr = [64, 34, 25, 12, 22, 11, 90]
insertionSort(arr)
print("Sorted array is:")
for i in range(len(arr)):
    print("% d" % arr[i], end=" ")
```

```
Sorted array is:
 11  12  22  25  34  64  90
```