Nama : Khoirul Anam

Nim : G.211.22.0082

Kelas : B1

- ## Algoritma Bubble Sort

1. Fungsi “bubble_sort(arr)” menerima satu parameter, yaitu sebuah array “arr” yang akan diurutkan.
2. Variabel “n” diinisialisasi dengan panjang array “arr”.
3. Dilakukan perulangan luar dengan menggunakan “for i in range(n)”. Perulangan ini akan melakukan iterasi sebanyak panjang array.
4. Di dalam perulangan luar, terdapat perulangan dalam (“for j in range(0, n-i-1)”). Perulangan dalam ini digunakan untuk membandingkan elemen-elemen berdekatan dalam array.
5. Pada setiap iterasi perulangan dalam, dilakukan pengecekan apakah elemen pada indeks ke-j lebih besar dari elemen pada indeks ke-(j+1). Jika ya, maka dilakukan pertukaran elemen tersebut.
Langkah ini dilakukan untuk memastikan bahwa elemen terbesar berada di akhir array setelah iterasi tersebut.
6. Setelah perulangan dalam selesai, dilakukan pengecekan apakah ada pertukaran elemen pada iterasi tersebut dengan menggunakan variabel “swapped”. Jika tidak ada pertukaran yang dilakukan, berarti array sudah terurut, dan perulangan luar dapat dihentikan dengan menggunakan pernyataan “break”.
Hal ini membantu meningkatkan efisiensi algoritma, karena jika array sudah terurut sebelum mencapai iterasi terakhir, kita tidak perlu melanjutkan iterasi.
7. Setelah seluruh iterasi selesai, array “arr” yang diinput telah diurutkan menggunakan algoritma Bubble Sort.
8. Pada akhir program, array awal (“arr”) dicopy ke array baru (“bubble_sort_arr”) untuk memastikan array awal tetap tidak terurut setelah pengurutan menggunakan Bubble Sort.
9. Hasil pengurutan dicetak menggunakan pernyataan “print”.

- ## Algoritma Selection Sort

1. Fungsi “selection_sort(arr)” menerima satu parameter, yaitu sebuah array “arr” yang akan diurutkan.
2. Variabel “n” diinisialisasi dengan panjang array “arr”.
3. Dilakukan perulangan luar dengan menggunakan “for i in range(n)”. Perulangan ini akan melakukan iterasi sebanyak panjang array
4. Di dalam perulangan luar, variabel “min_index” diinisialisasi dengan nilai “I”. Variabel ini akan digunakan untuk menyimpan indeks elemen terkecil dalam array yang belum diurutkan.
5. Terdapat perulangan dalam (“for j in range(i+1, n)”) yang dimulai dari indeks berikutnya setelah “I”. Perulangan ini digunakan untuk mencari elemen terkecil dalam array yang belum diurutkan.

6. Pada setiap iterasi perulangan dalam, dilakukan pengecekan apakah elemen pada indeks “j” lebih kecil dari elemen pada indeks “min_index”. Jika ya, maka nilai “min_index” diupdate menjadi “j”.
Langkah ini memastikan bahwa kita menemukan elemen terkecil dalam array yang belum diurutkan.
7. Setelah perulangan dalam selesai, dilakukan pertukaran elemen antara elemen pada indeks “I” dengan elemen terkecil yang ditemukan (“min_index”).
Langkah ini memastikan bahwa elemen terkecil ditempatkan pada posisi yang tepat dalam array yang sudah diurutkan.
8. Setelah seluruh iterasi selesai, array “arr” yang diinput telah diurutkan menggunakan algoritma Selection Sort.
9. Pada akhir program, array awal (“arr”) dicopy ke array baru (“selection_sort_arr”) untuk memastikan array awal tetap tidak terurut setelah pengurutan menggunakan Selection Sort.
10. Hasil pengurutan dicetak menggunakan pernyataan “print”.

- **Algoritma Insertion Sort**

1. Fungsi “insertion_sort(arr)” menerima satu parameter, yaitu sebuah array “arr” yang akan diurutkan.
2. Dilakukan perulangan luar dengan menggunakan “for i in range(1, len(arr))”. Perulangan ini akan melakukan iterasi sebanyak panjang array, dimulai dari indeks kedua.
3. Di dalam perulangan luar, nilai pada indeks ke-i disimpan dalam variabel “key”. Variabel ini akan menjadi elemen yang akan "dimasukkan" ke dalam bagian yang sudah diurutkan dari array.
4. Variabel “j” diinisialisasi dengan nilai “i-1”. Perulangan dalam (“while j >= 0 and key < arr[j]”) akan berjalan selama “j” tidak kurang dari 0 dan nilai “key” lebih kecil dari elemen pada indeks ke-“j”.
5. Di dalam perulangan dalam, dilakukan pergeseran elemen-elemen ke posisi yang lebih tinggi untuk memberikan ruang bagi “key”.
Langkah ini akan dilakukan selama “key” lebih kecil dari elemen pada indeks ke-“j”.
6. Setelah keluar dari perulangan dalam, “key” ditempatkan pada posisi yang sesuai dalam array yang sudah diurutkan.
7. Setelah seluruh iterasi selesai, array “arr” yang diinput telah diurutkan menggunakan algoritma Insertion Sort.
8. Pada akhir program, array awal (“arr”) dicopy ke array baru (“insertion_sort_arr”) untuk memastikan array awal tetap tidak terurut setelah pengurutan menggunakan Insertion Sort.
9. Hasil pengurutan dicetak menggunakan pernyataan “print”.