TEORI 1

# ALGORITMA SORTING

Disusun untuk Memenuhi Matakuliah Algoritma Struktur Data

Dosen Pengampu : Dr Arna Fariza S.Kom., M.Kom.


Oleh :

Nama Ghaly Abrarian

NRP : 31235000

Kelas : D3 IT A

Politeknik Elektronika Negeri Surbaya

Departemen Teknik Informatika dan Komputer

Program Studi Teknik Informatika

## A. SOAL

Sekumpulan bilangan acak berikut:

58 78 97 48 67 86 6 57 76 95

Tuliskan Langkah-langkah pada outer loop dengan algoritma pengurutan

a. Insertion

b. Selection

c. Bubble

d. Bubble Flag

e. Shell

f. Quick

g. Merge

## B. JAWABAN

1. Insertion Sort

```
// Insertion Sort
      58 78 97 48 67 86 6  57 76 95
i=0   58 78 97 48 67 86 6  57 76 95
i=1   58 78 97 48 67 86 6  57 76 95
i=2   58 78 97 48 67 86 6  57 76 95
i=3   48 58 78 97 67 86 6  57 76 95
i=4   48 58 67 78 97 86 6  57 76 95
i=5   48 58 67 78 86 97 6  57 76 95
i=6   6  48 58 67 78 86 97 57 76 95
i=7   6  48 57 58 67 78 86 97 76 95
i=8   6  48 57 58 67 76 78 86 97 95
i=9   6  48 57 58 67 76 78 86 95 97
// Data bewarna kuning adalah data yang sedang dibandingkan
```

1. i=0 bernilai 58 dianggap sudah urut
2. i=1 bernilai 78 dibandingkan dengan 58. Tidak perlu penggeseran karena sudah dalam urutan.
3. i=2 bernilai 97 dibandingkan dengan 78. Tidak perlu penggeseran.
4. i=3 bernilai 48 dibandingkan dengan 97, 78, dan 58. Penggeseran terjadi karena 48 lebih kecil dari data yang sudah diurutkan sebelumnya dan 48 ditempatkan pada posisi pertama.
5. i=4 bernilai 67 dibandingkan dengan 97, 78, dan 58. 67 ditempatkan diantara 58 dan 78 dikarenakan 67 lebih besar dari 58 dan lebih kecil dari 78.

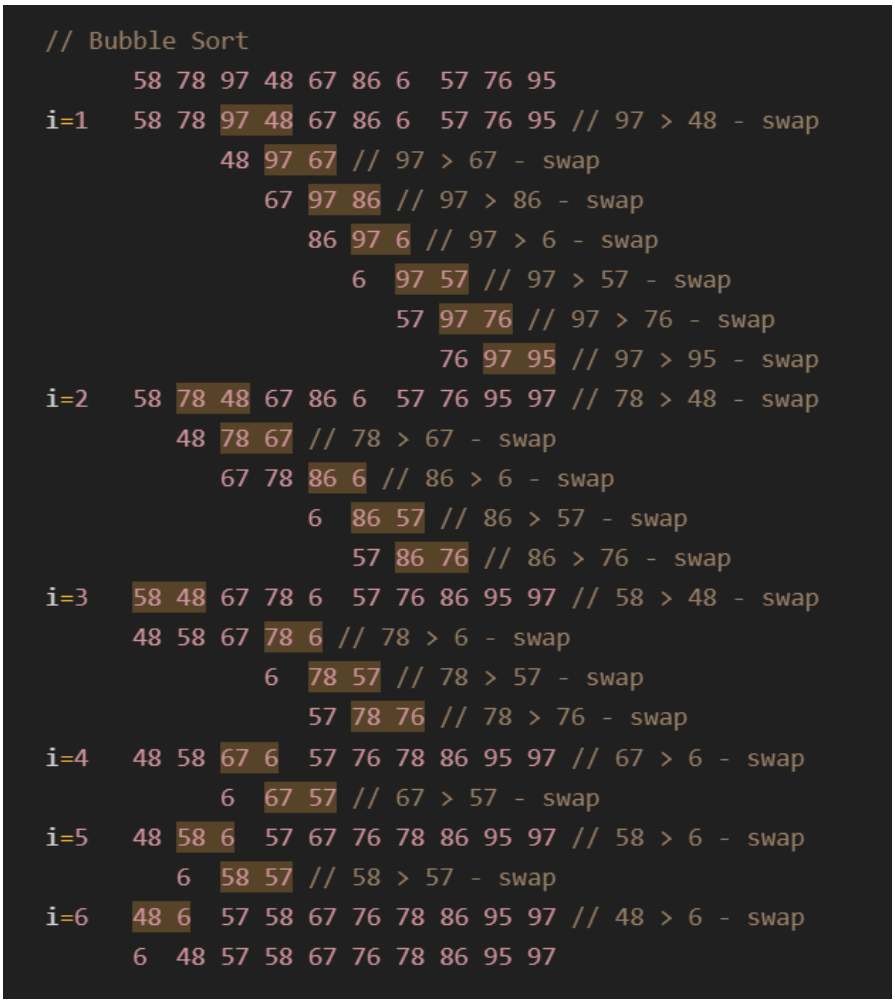
6. i=5 bernilai 86 dibandingkan dengan 97 dan 78. 86 ditempatkan diantara 78 dan 97 karena 86 lebih besar dari 78 dan lebih kecil dari 97.
7. i=6 bernilai 6 dibandingkan dengan semua elemen sebelumnya. 6 ditempatkan di posisi pertama karena nilai yang paling kecil dari seluruh data yang sudah dibandingkan.
8. i=7 bernilai 57 dibandingkan dengan semua elemen sebelumnya. Ditempatkan pada posisi setelah 48 karena 57 lebih besar dari 48, dan sebelum 58 karena 57 lebih kecil dari 58
9. i=8 bernilai 76 dibandingkan dengan elemen sebelumnya. Ditempatkan pada posisi setelah 67 dan sebelum 78
10. i=9 bernilai 95 ditempatkan pada posisi setelah 95 dan sebelum 97.
11. data sudah terurutkan.

2. Selection Sort

```
// Selection Sort
      58 78 97 48 67 86 6  57 76 95
i=0   58 78 97 48 67 86 6  57 76 95
i=1   6  78 97 48 67 86 58 57 76 95
i=2   6  48 97 78 67 86 58 57 76 95
i=3   6  48 57 78 67 86 58 97 76 95
i=4   6  48 57 58 67 86 78 97 76 95
i=5   6  48 57 58 67 86 78 97 76 95
i=6   6  48 57 58 67 76 78 97 86 95
i=7   6  48 57 58 67 76 78 97 86 95
i=8   6  48 57 58 67 76 78 86 97 95
i=9   6  48 57 58 67 76 78 86 95 97
```

1. Outer loop pertama: Temukan elemen terkecil (6) dan tukar dengan elemen pertama (58).
2. Outer loop kedua: Temukan elemen terkecil (48) dan tukar dengan elemen kedua (78).
3. Outer loop ke tiga: Temukan elemen terkecil (57) dan tukar dengan elemen ketiga (97).
4. Outer loop ke empat: Temukan elemen terkecil (58) dan tukar dengan elemen keempat (78).
5. Outer loop ke lima: Temukan elemen terkecil (67), tidak perlu ditukar karena tidak ada elemen yang lebih kecil lagi dari 67
6. Outer loop ke enam: Temukan elemen terkecil (76) dan tukar dengan elemen keenam (86).
7. Outer loop ke tujuh: Temukan elemen terkecil (78), tidak perlu ditukar karena tidak ada elemen yang lebih kecil lagi dari 78
8. Outer loop ke delapan: Temukan elemen terkecil (86) dan tukar dengan elemen kedelapan (97).
9. Outer loop ke sembilan: Temukan elemen terkecil (95) dan tukar dengan elemen kedelapan (97).
10. data sudah terurut.

3. Bubble Sort

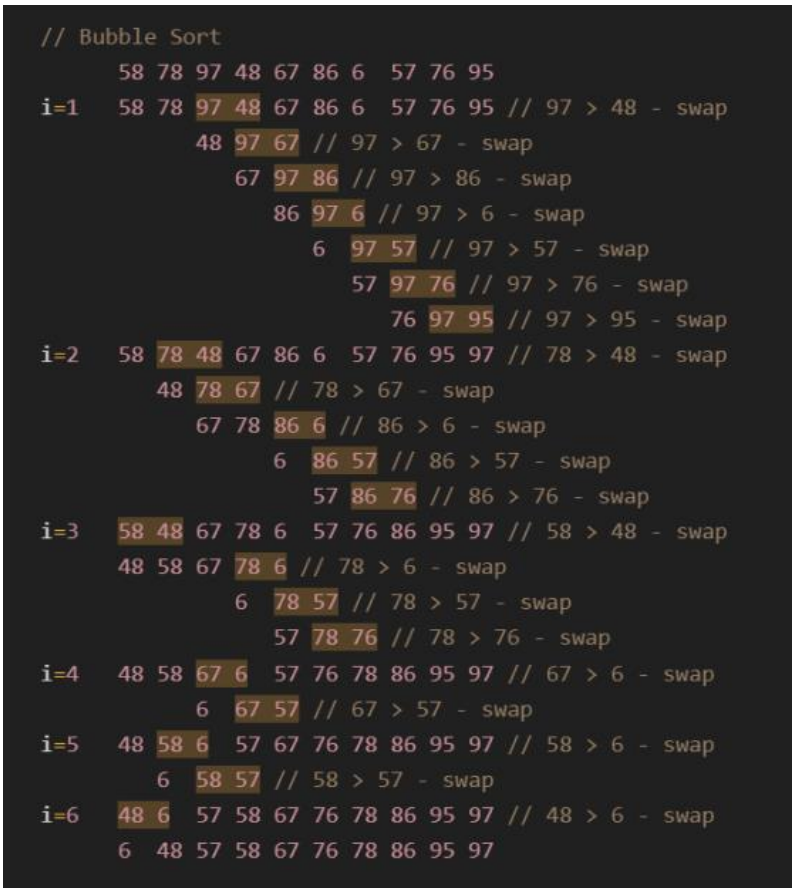



1. i=1, looping pertama, nilai 97 di swap oleh beberapa nilai hingga ke urutan terakhir, karena nilai data 97 merupakan yang paling besar diantara nilai data yang lain
2. i=2, looping kedua, nilai data 78 di swap 2 kali dengan 48 dan 67 karena 78 lebih besar dari 48 dan 67. Nilai data 86 di swap oleh 6, 57, 76 karena nilai data 86 lebih besar dari ketiga nilai data tersebut.
3. i=3. looping ketiga, 58 di swap dengan 48 karena 58 lebih besar dari 48. Nilai data 78 di swap oleh 6, 57, 76 karena nilai data 86 lebih besar dari ketiga nilai data tersebut.
4. i=4, looping keempat, nilai data 67 di swap dengan nilai data 6 dan 57 karena nilai data 67 lebih besar dari kedua nilai data tersebut.
5. i=5, looping kelima, nilai data 58 di swap dengan nilai data 6 dan 57 karena nilai data 58 lebih besar dari kedua nilai data tersebut.
6. i=6, looping keenam, nilai data 48 di swap dengan 6 karena 48 lebih besar dari 6.
7. data sudah terurut.

Analisa:

Algoritma diatas berjalan mulai dari kiri ke kanan, algortima bubble membaca 2 data dan membandingkannya. Apabila data sebelum lebih besar daripada data sesudah maka posisi kedua data tersebut akan ditukar, begitu juga sebaliknya apabila data sebelum lebih kecil daripada data sesudah, maka tidak akan ada proses penukaran posisi. Proses ini berjalan sampai proses pembacaaan dan perbandingan 2 data sudah mencapai akhir dari baris data. Saat sudah tidak ada data yang perlu ditukar/diurutkan, maka proses akan selesai.

4. Bubble Flag



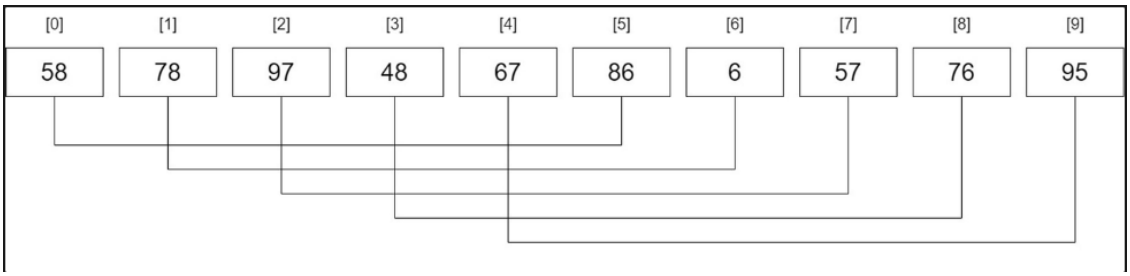

1. i=1, looping pertama, nilai 97 di swap oleh beberapa nilai hingga ke urutan terakhir, karena nilai data 97 merupakan yang paling besar diantara nilai data yang lain
2. i=2, looping kedua, nilai data 78 di swap 2 kali dengan 48 dan 67 karena 78 lebih besar dari 48 dan 67. Nilai data 86 di swap oleh 6, 57, 76 karena nilai data 86 lebih besar dari ketiga nilai data tersebut.
3. i=3. looping ketiga, 58 di swap dengan 48 karena 58 lebih besar dari 48. Nilai data 78 di swap oleh 6, 57, 76 karena nilai data 86 lebih besar dari ketiga nilai data tersebut.
4. i=4, looping keempat, nilai data 67 di swap dengan nilai data 6 dan 57 karena nilai data 67 lebih besar dari kedua nilai data tersebut.
5. i=5, looping kelima, nilai data 58 di swap dengan nilai data 6 dan 57 karena nilai data 58 lebih besar dari kedua nilai data tersebut.
6. i=6, looping keenam, nilai data 48 di swap dengan 6 karena 48 lebih besar dari 6.
7. data sudah terurut.

Analisa:

Algoritma diatas berjalan mulai dari kiri ke kanan, algortima bubble membaca 2 data dan membandingkannya. Apabila data sebelum lebih besar daripada data sesudah maka posisi kedua data tersebut akan ditukar, begitu juga sebaliknya apabila data sebelum lebih kecil daripada data sesudah, maka tidak akan ada proses penukaran posisi. Proses ini berjalan sampai proses pembacaaan dan perbandingan 2 data sudah mencapai akhir dari baris data. Saat sudah tidak ada data yang perlu ditukar/diurutkan, maka proses akan selesai.

Algoritma ini seperti bubble sort, tetapi memiliki flag yang menunjukkan apakah pertukaran terjadi atau tidak. Jika tidak ada pertukaran dalam satu putaran loop, pengurutan dianggap selesai.

5. Shell Sort




Sublist k=5

S[0] < S[5] OK

S[1] > S[6] → 78 > 6 → swap

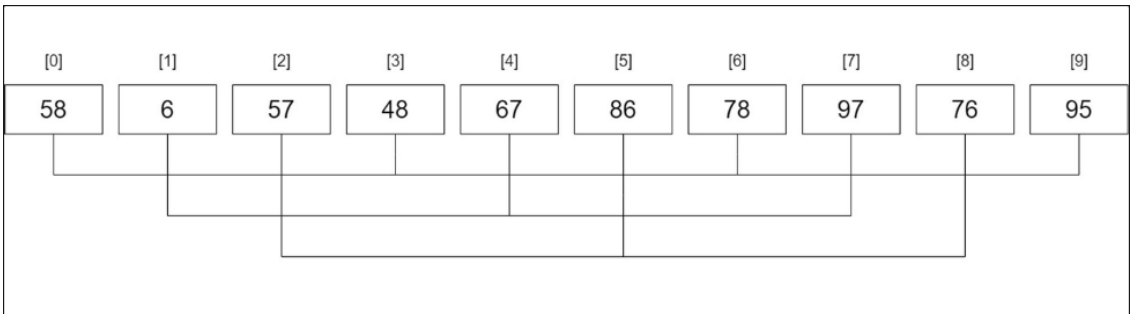
S[2] > S[7] → 97 > 57 → swap

S[3] < S[8] OK

S[4] < S[9] OK

Analisa:

Pertukaran data pada jarak 5 pada masing-masing data.




Sublist k=3

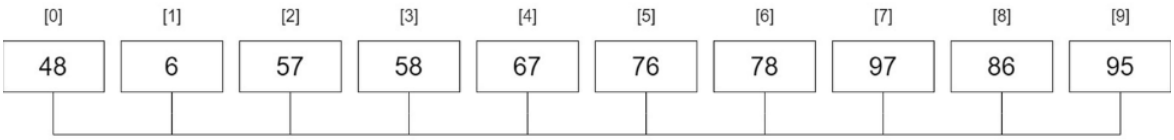
S[0] S[3] S[6] S[9] → 58, 48, 78, 95 → Not OK → Sort → 48, 58, 78, 95

S[1] S[4] S[7] → 6, 67, 97 → OK

S[2] S[5] S[8] → 57, 86, 76 → Not OK → Sort → 57, 76, 86

Analisa:

Pertukaran data pada jarak 3 pada masing-masing data.




Sublist k=1

Sorting seperti insertion sort

→ 6 48 57 58 67 76 78 86 95 97

6 swap dengan 48

97 swap dengan 86

97 swap dengan 95

Analisa:

Pertukaran data pada jarak 1 pada masing-masing data.

6. Quick Sort

1. Pilih elemen pivot (misalnya, elemen tengah)
2. Bagi array menjadi 2 bagian, satu dengan elemen yang lebih kecil dari pivot, dan satu dengan elemen yang lebih besar.
3. Lakukan rekursi pada kedua bagian array

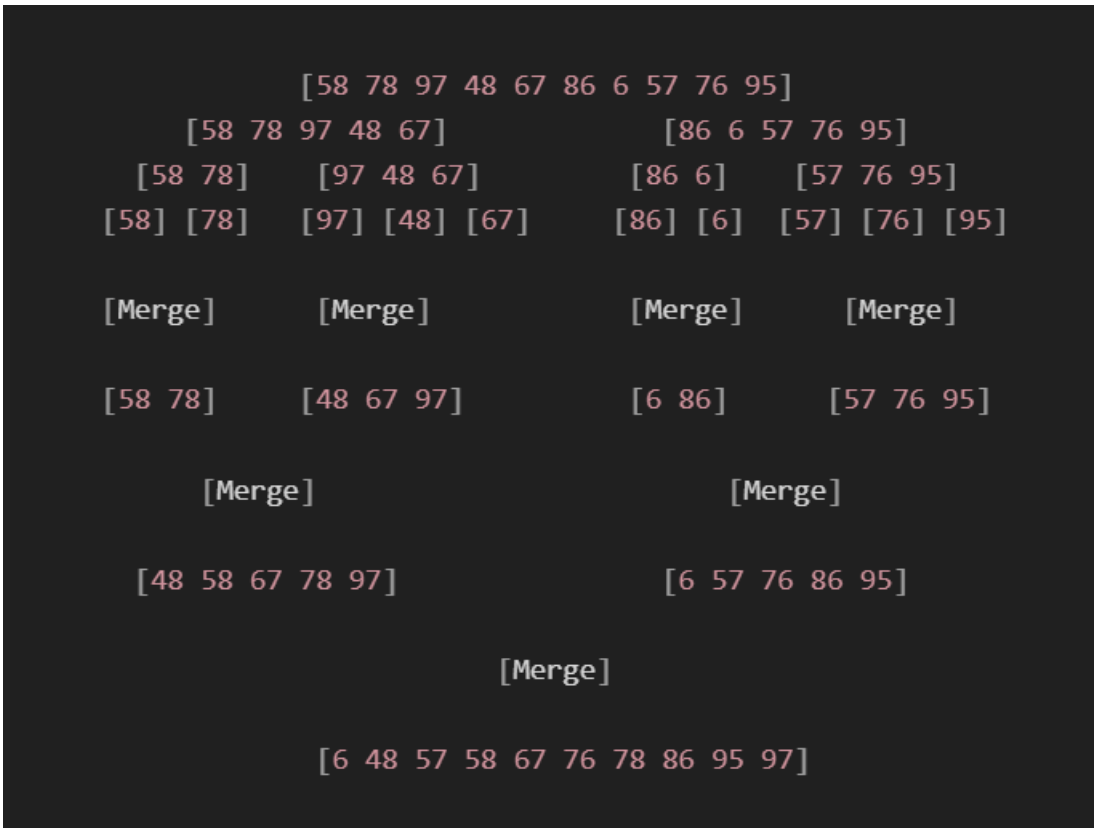
Langkah-langkah di setiap iterasi:

```
// Iterasi 1
[48 57 6 58 67] [76 78 86 95 97]

// Iterasi 2
[6 48 57 58 67] [76 78 86 95 97]

// Iterasi 3
[6 48 57 58 67 76 78 86 95 97]
```

7. Merge Sort




Merge Sort merupakan algoritma divide-and-conquer (membagi dan menyelesaikan). Membagi array menjadi dua bagian sampai subarray hanya berisi satu elemen. Seperti pada percobaan diatas, data yang masih acak dipecah-pecah sampai mereka berdiri sendiri ( 1 kotak 1 data ). Setelah itu, dilakukan proses merge, yaitu proses menggabungkan dan mengurutkan data-data yang sudah dipecah tadi, sehingga menghasilkan suatu baris data yang sudah urut. Proses merge dilakukan sampai semua data menjadi 1 baris gabungan data.