

Algoritma Sorting (Selection – Insertion)

Algoritma Insertion Sort

- Dengan Algoritma Insertion bagian kiri array terurut sampai seluruh array
- Misal pada data array ke-k, data tersebut akan disisipkan pada indeks sebelum k, sesuai dengan urutannya. Proses ini dilakukan berulang-ulang sehingga seluruh data terurut

Algoritma Insertion Sort

1. $i \leftarrow 1$

2. selama ($i < N$) kerjakan baris 3 sampai dgn 9

3. $key \leftarrow A[i]$


4. $j \leftarrow i - 1$

C = Comparison




5. selama $j \geq 0$ dan ($A[j] > key$) kerjakan baris 6 dan 7

6. $A[j + 1] \leftarrow A[j]$ ← M = Move



7. $j \leftarrow j - 1$

8. $A[j+1] \leftarrow key$ ← M = Move



9. $i \leftarrow i + 1$

Proses Insertion Sort

3	10	4	6	8	9	7	2	1	5
---	----	---	---	---	---	---	---	---	---

Nilai yang paling kiri (3) terurut terhadap dirinya sendiri. Sehingga kita tidak perlu melakukan apa-apa

3	10	4	6	8	9	7	2	1	5
----------	----	---	---	---	---	---	---	---	---

3	10	4	6	8	9	7	2	1	5
---	-----------	---	---	---	---	---	---	---	---

Key = 10

Cek apakah nilai pada indek ke-2 (10) lebih kecil dari indek ke-1 (3). Jika ya maka tukar. Kondisi diatas maka data tidak perlu diswap

3	10	4	6	8	9	7	2	1	5
---	----	---	---	---	---	---	---	---	---

Dua data pertama sudah terurut

3	10	4	6	8	9	7	2	1	5
---	----	---	---	---	---	---	---	---	---

Selanjutnya menyisipkan 4 di daerah abu-abu sehingga setelah penyisipan daerah abu-abu menjadi urut.

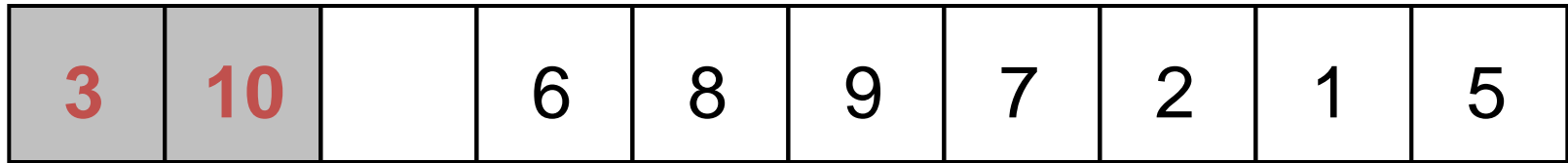
Simpan nilai yang akan diurutkan dalam variabel key

Key = 4

C

4

C = Comparison

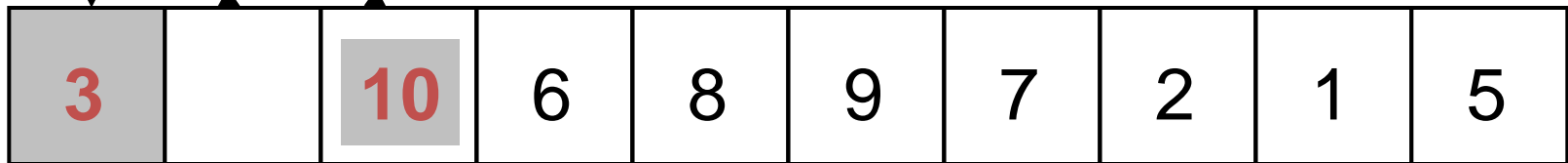


Nilai 10 digeser pada indek ke-3

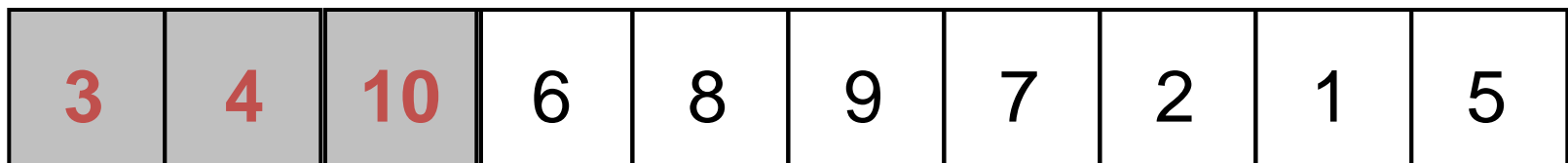
M

4

M = Move



Sisipkan 4 pada posisi yang tepat



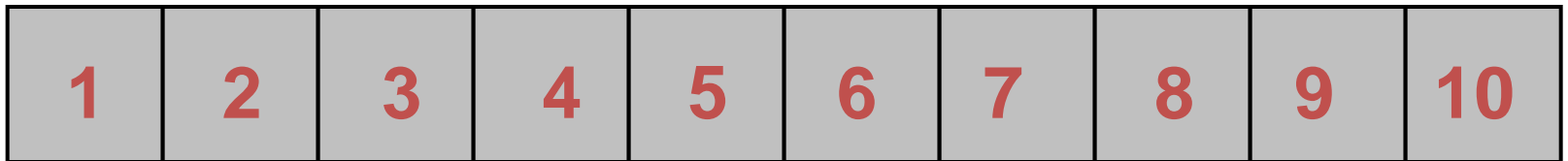
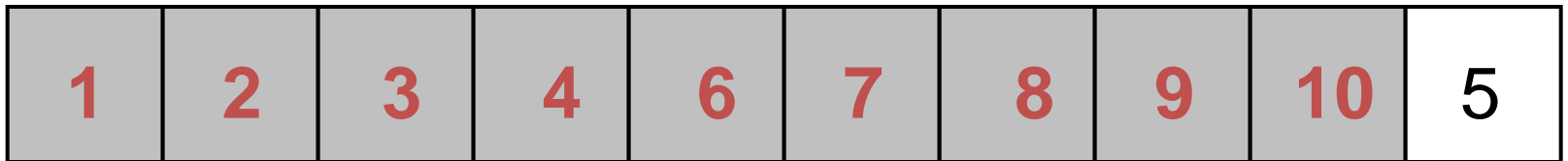
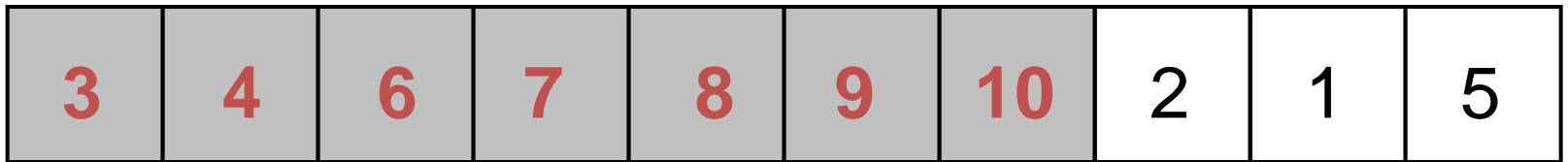
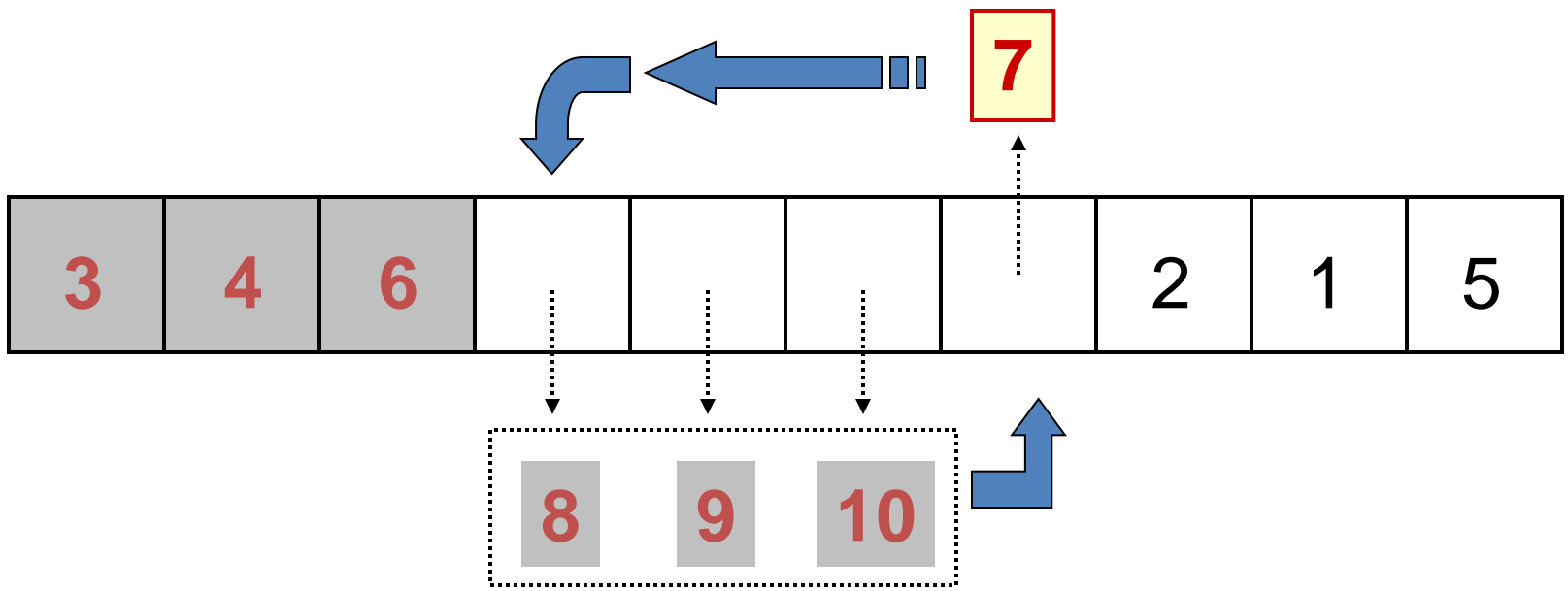
Sekarang 3 data pertama sudah terurut secara relatif dan sisipkan data ke-4 ke daerah abu-abu. 4 Data pertama sudah terurut secara relatif

3	4	6	10	8	9	7	2	1	5
---	---	---	----	---	---	---	---	---	---

Ulangi Proses ini sampai data terurut semuanya.

3	4	6	8	10	9	7	2	1	5
---	---	---	---	----	---	---	---	---	---

3	4	6	8	9	10	7	2	1	5
---	---	---	---	---	----	---	---	---	---



Analisa Insertion Sort

- Basic Operation (Operasi Dasar) terdapat pada perbandingan key
($A[j] > \text{key}$)

Analisa Insertion Sort

Best Case

- Array sudah dalam keadaan terurut naik
- Data ke- k yang akan diurutkan dibandingkan sebanyak satu kali dengan data ke- $(k-1)$
- Loop terdalam tidak pernah dieksekusi
- Jumlah pergeseran (*movement*) $\rightarrow M = 0$
- Jumlah pembandingan key (*comparison*)

$$C = n - 1$$

Worst Case:

- Array dalam urutan kebalikannya
- Loop terdalam dieksekusi sebanyak $p-1$ kali, untuk $p = 2, 3, \dots, n$
- Jumlah pergeseran

$$M = (n-1) + (1 + 2 + \dots + n-1)$$

$$M = (n-1) + n * (n-1) / 2$$

- Jumlah perbandingan key

$$C = (1 + 2 + \dots + n-1) = n * (n-1) / 2$$

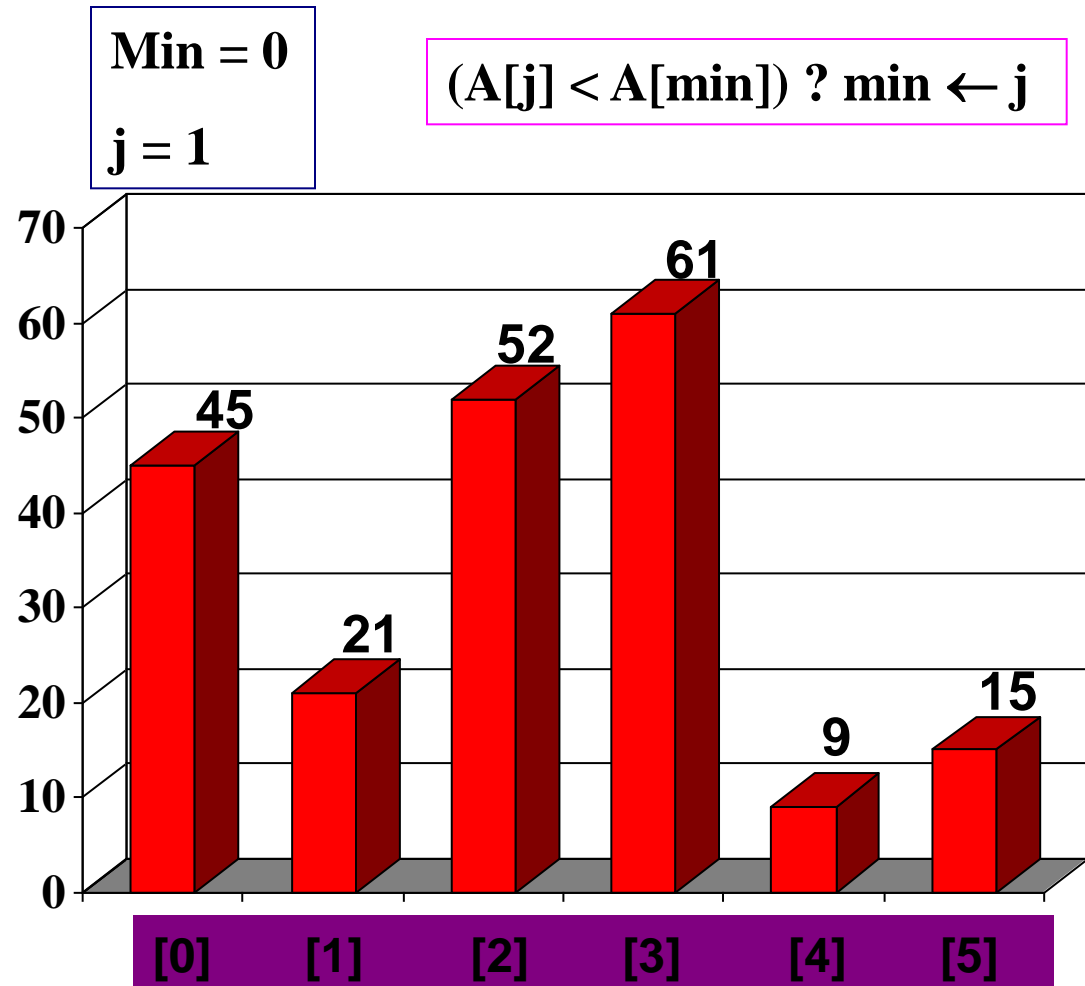
Selection Sort

Algoritma Selection Sort

1. $i \leftarrow 0$
2. selama ($i < N-1$) kerjakan baris 3 sampai dengan 11
3. $min \leftarrow i$
4. $j \leftarrow i + 1$
5. Selama ($j < N$) kerjakan baris 6 dan 7
6. Jika ($A[j] < A[min]$) maka $min \leftarrow j$
7. $j \leftarrow j + 1$
8. $temp = A[i]$
9. $A[i] = A[min]$
10. $A[min] = temp$
11. $i \leftarrow i + 1$

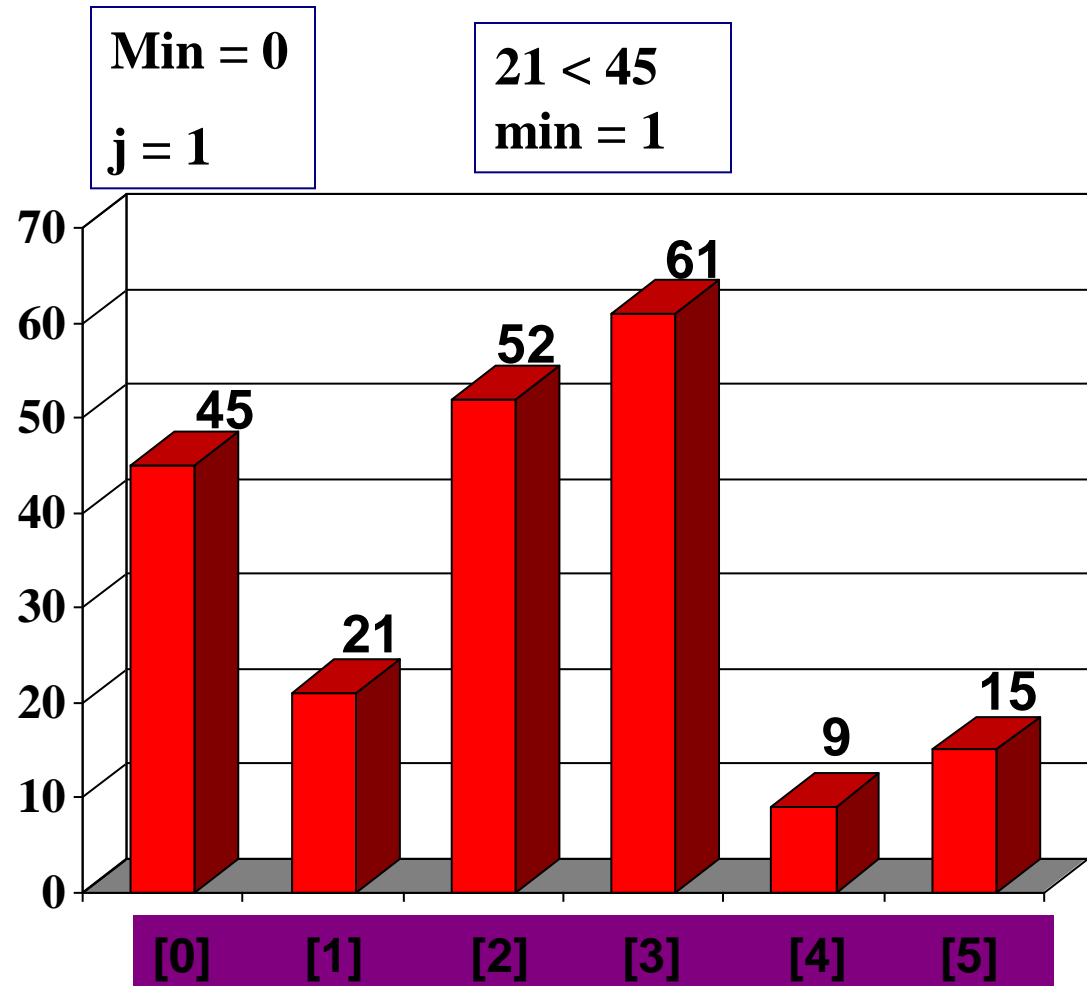
Selection Sort Algorithm

Perbandingan diawali pada data indeks ke-0. Dibandingkan dengan data sesudahnya untuk mencari elemen yang paling kecil (menggunakan indeks data)



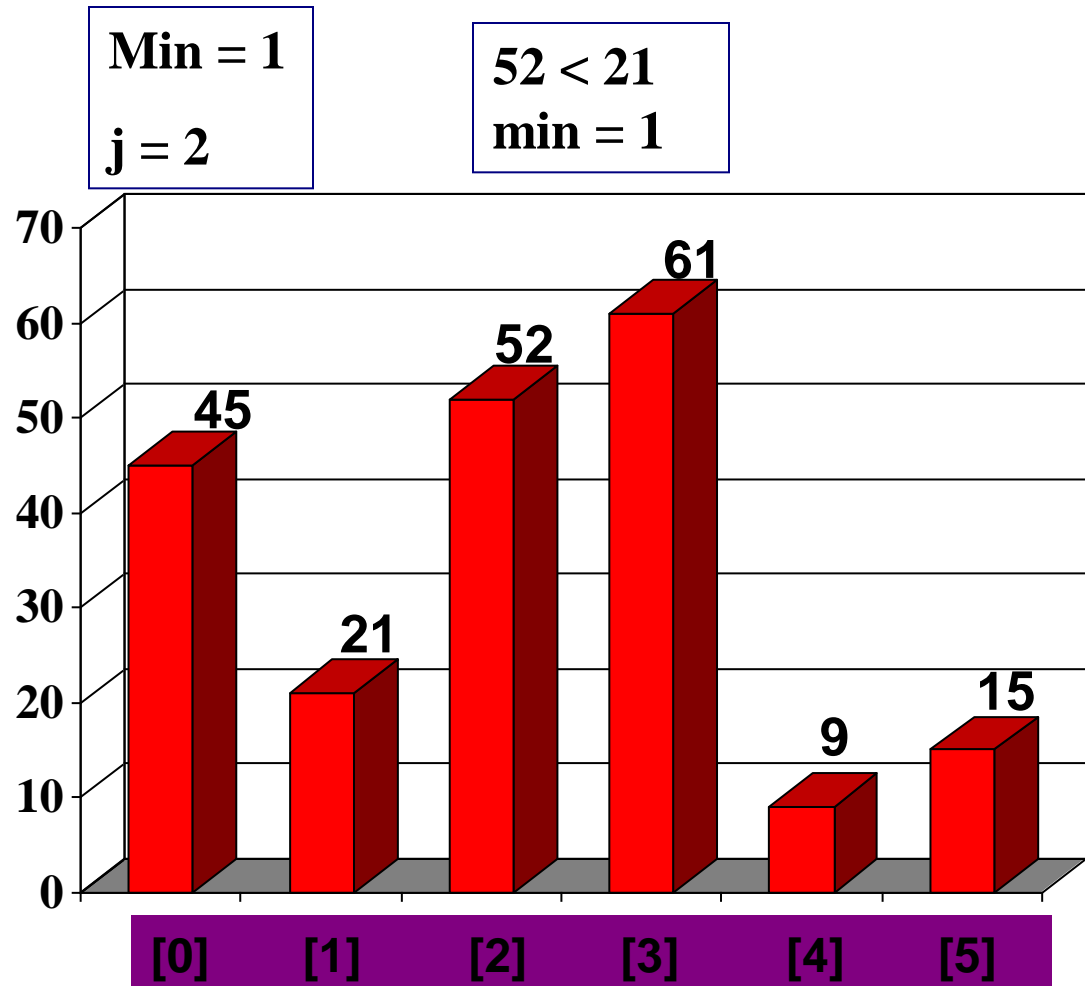
Selection Sort Algorithm – 1

Data ke-0
dibandingkan
dengan data
ke-1.
Ternyata
data min
pada indeks
ke-1
(min=1)



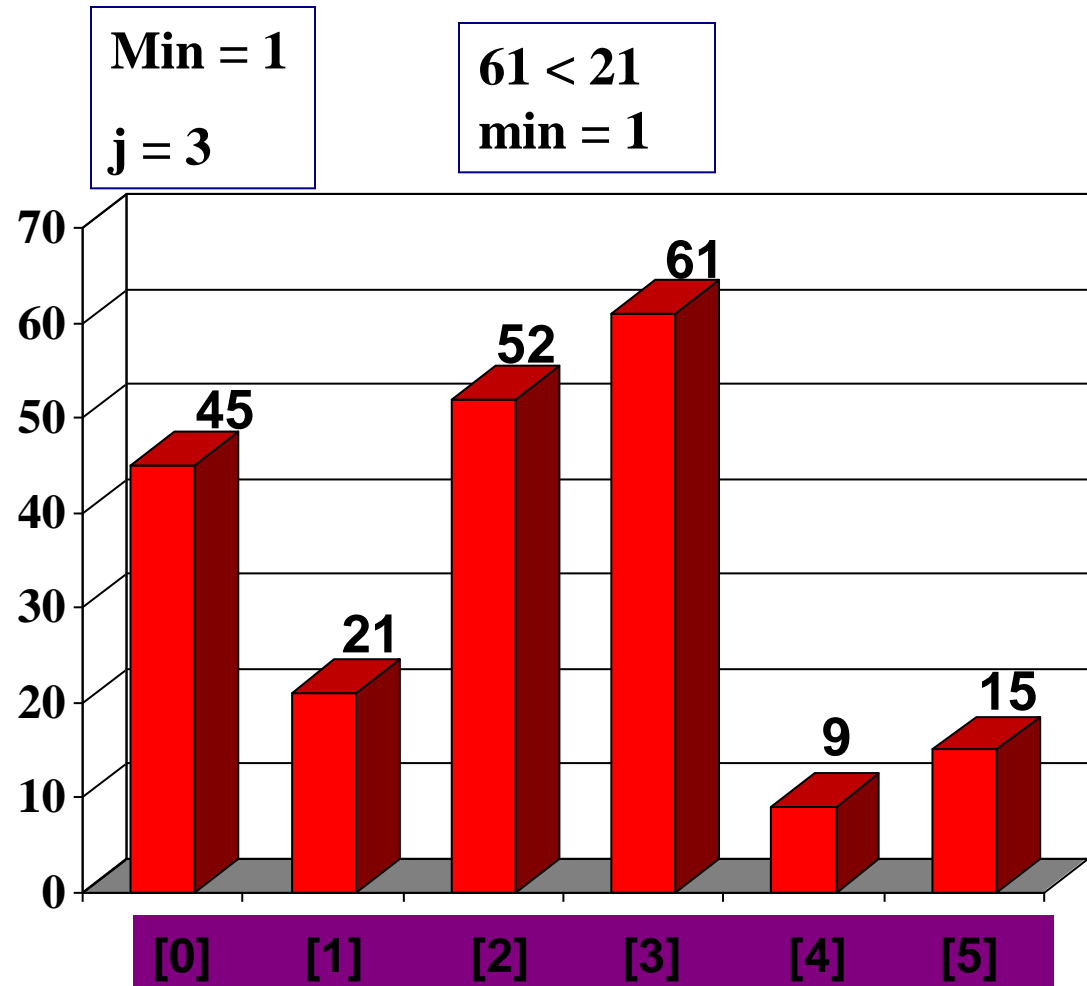
Selection Sort Algorithm – 2

Data ke1 dibandingkan dengan Data ke-2. Nilai yang paling kecil tetap pada data indeks ke-1



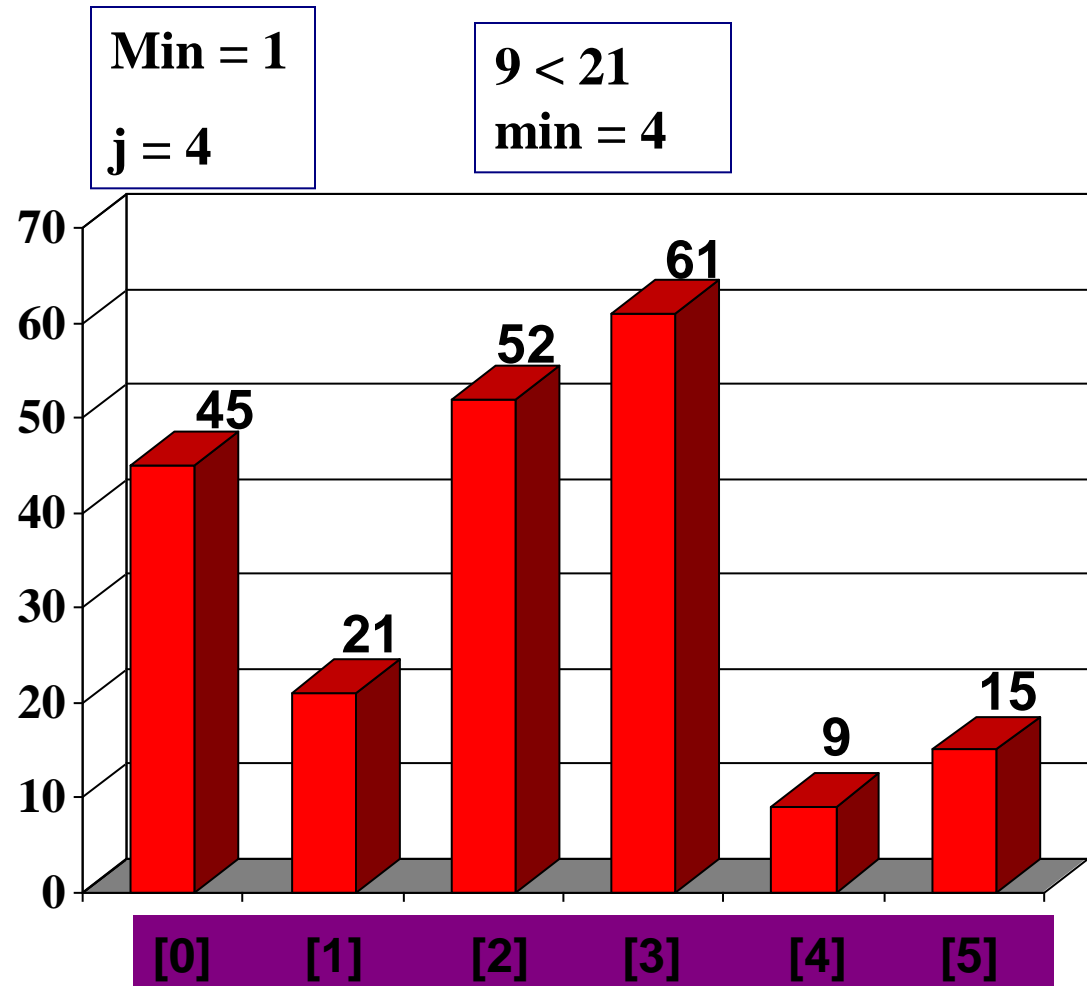
Selection Sort Algorithm – 3

Data ke1
dibandingkan
dengan Data
ke-3



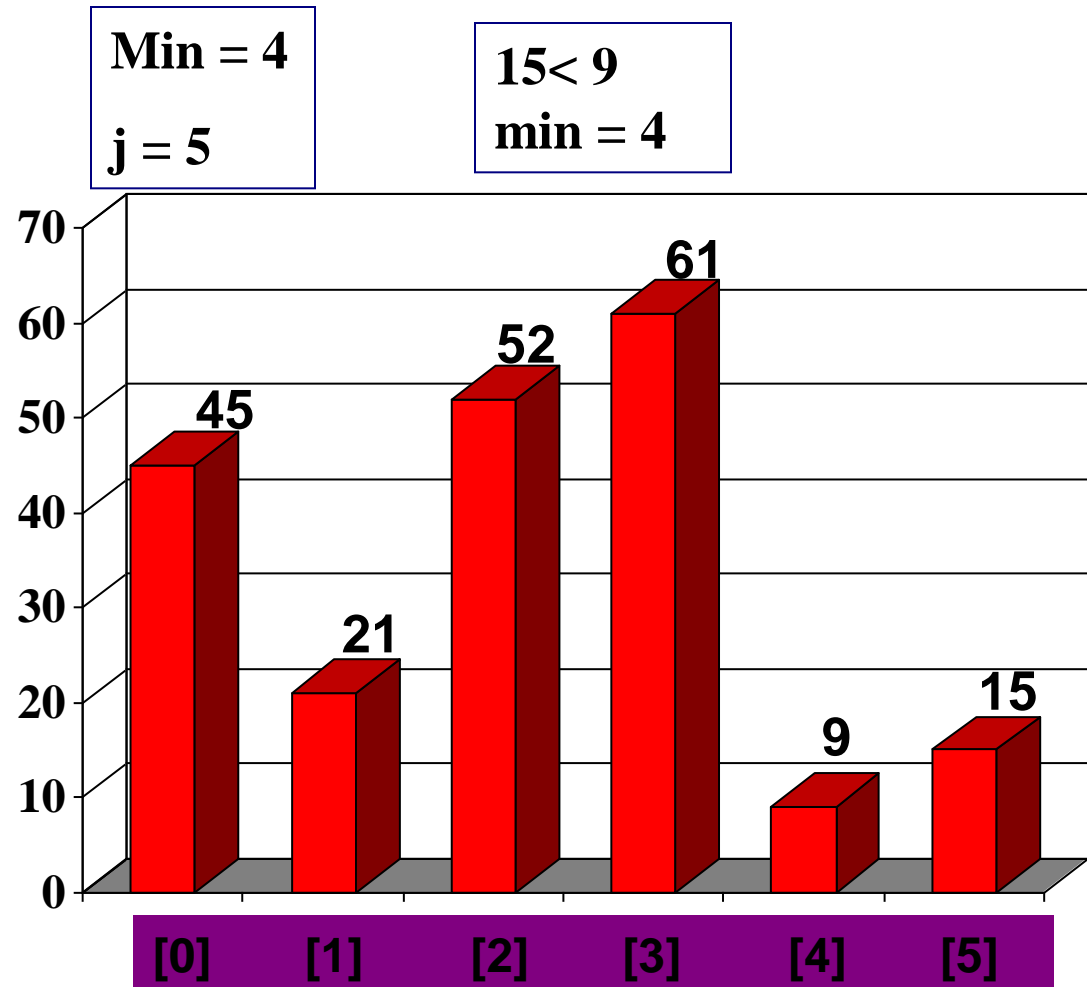
Selection Sort Algorithm – 4

Data ke1 dibandingkan dengan Data ke-4. Data yang paling kecil sekarang pada data indeks ke-4 (min =4)



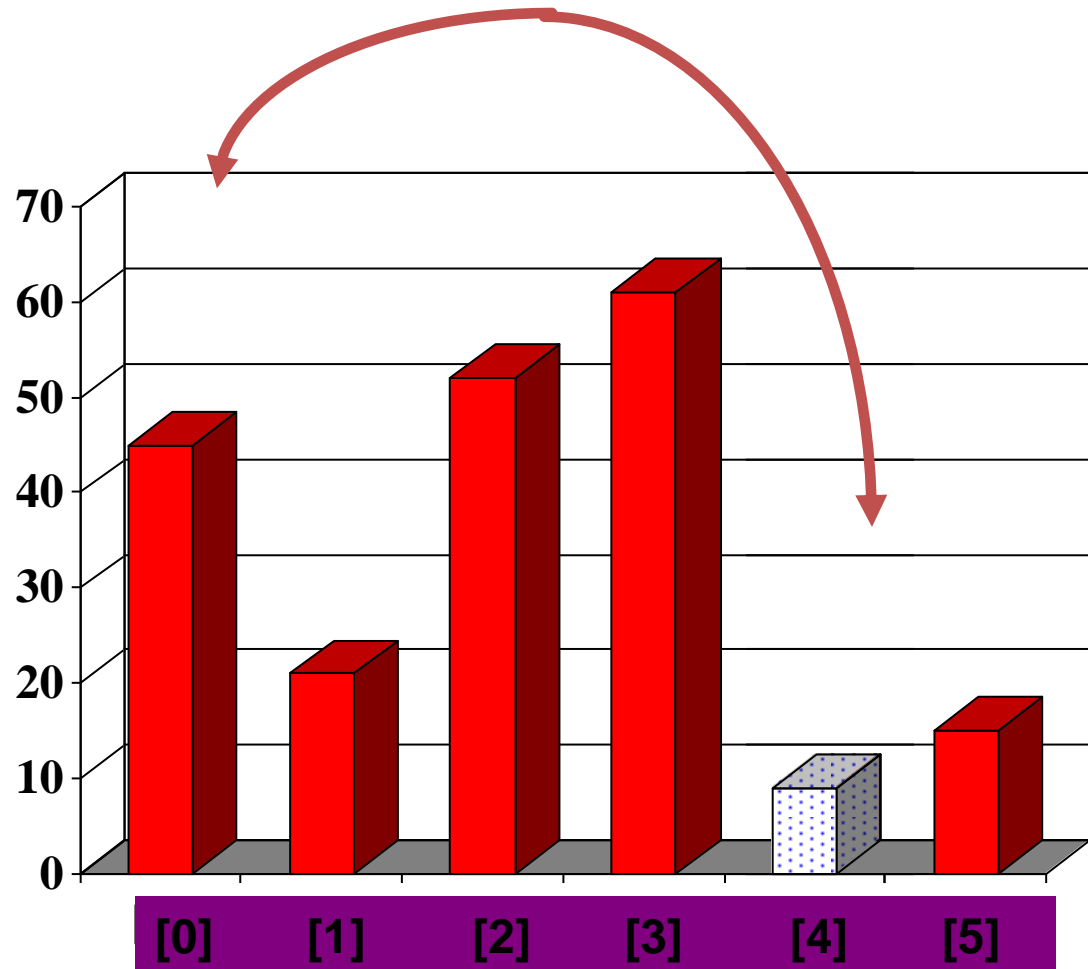
Selection Sort Algorithm – 5

Data ke-4
dibandingkan
dengan Data
ke-5



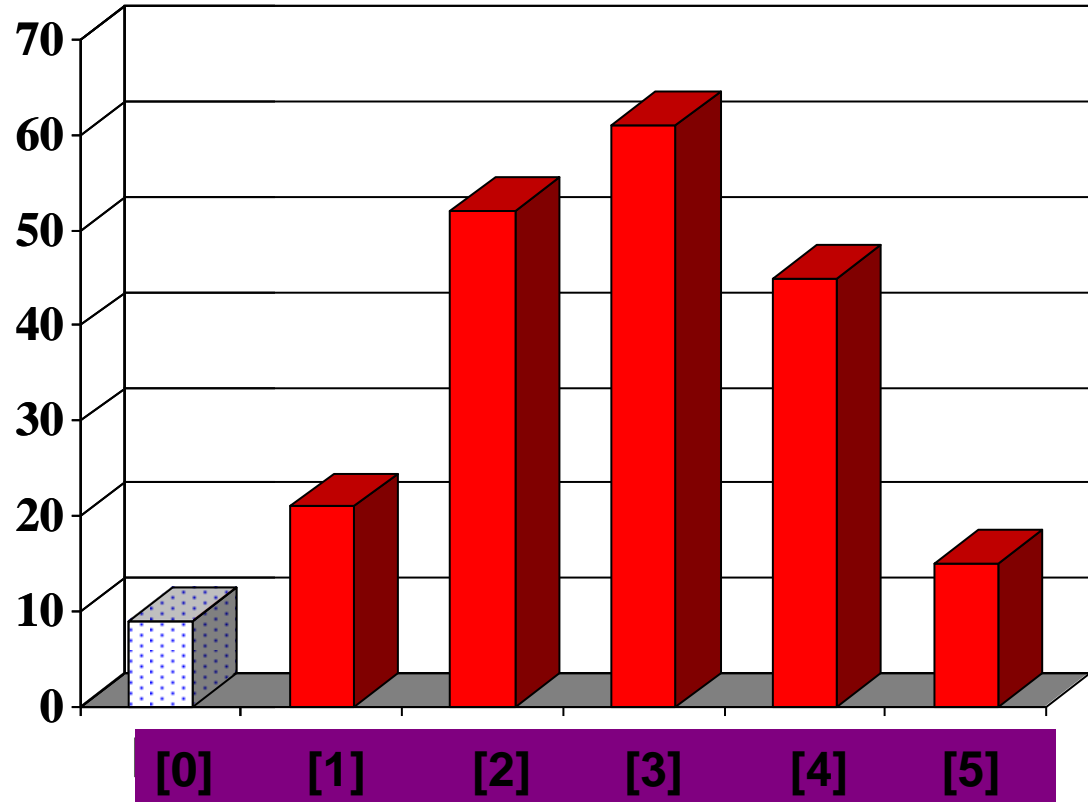
Selection Sort Algorithm – 2

- Diawali dengan mencari elemen yang paling kecil
- Tukar elemen yang paling kecil dengan Data indeks ke-0



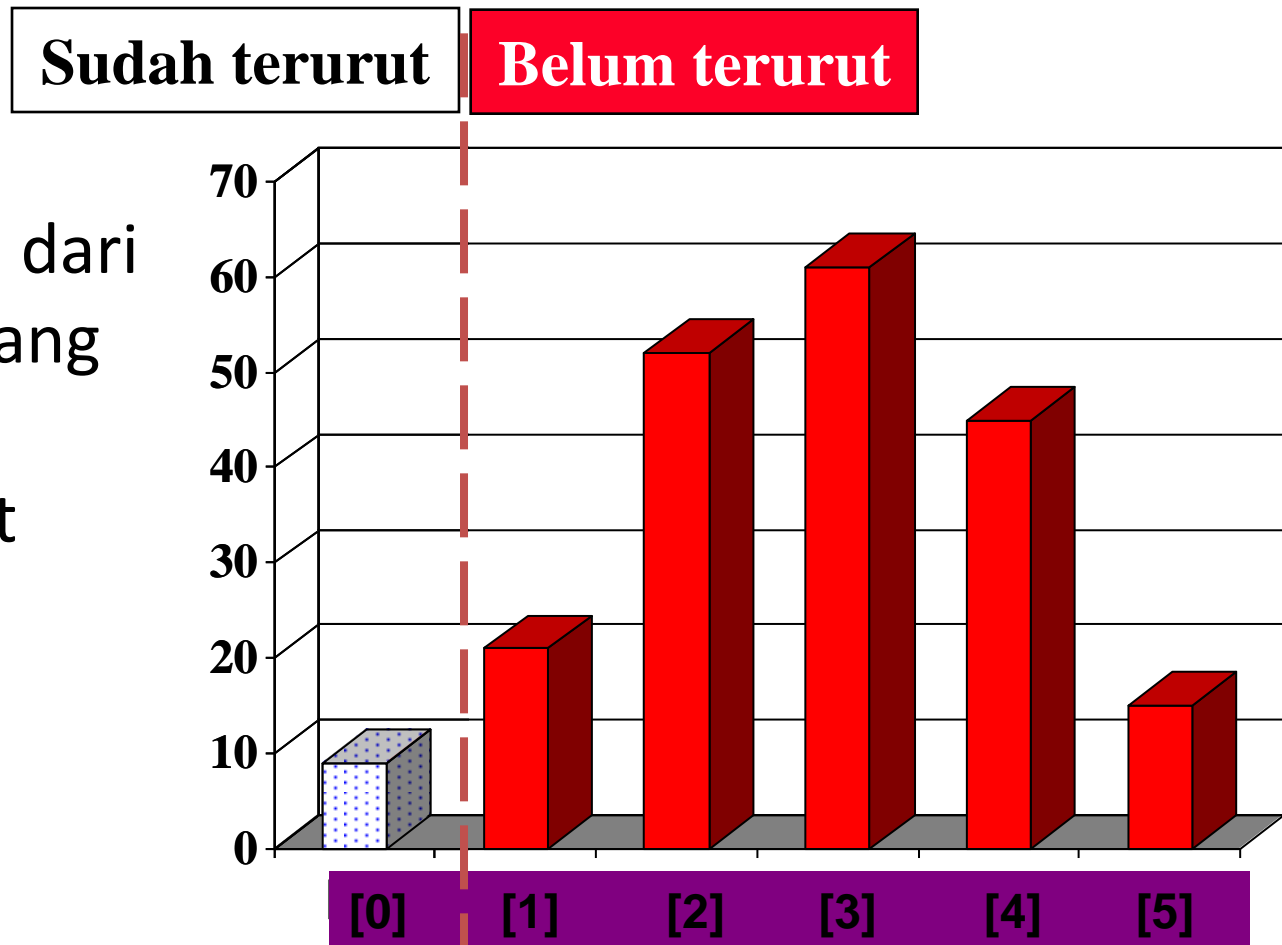
Selection Sort Algorithm – 3

- Diawali dengan mencari elemen yang paling kecil
- Tukar elemen yang paling kecil dengan Data indeks ke-0

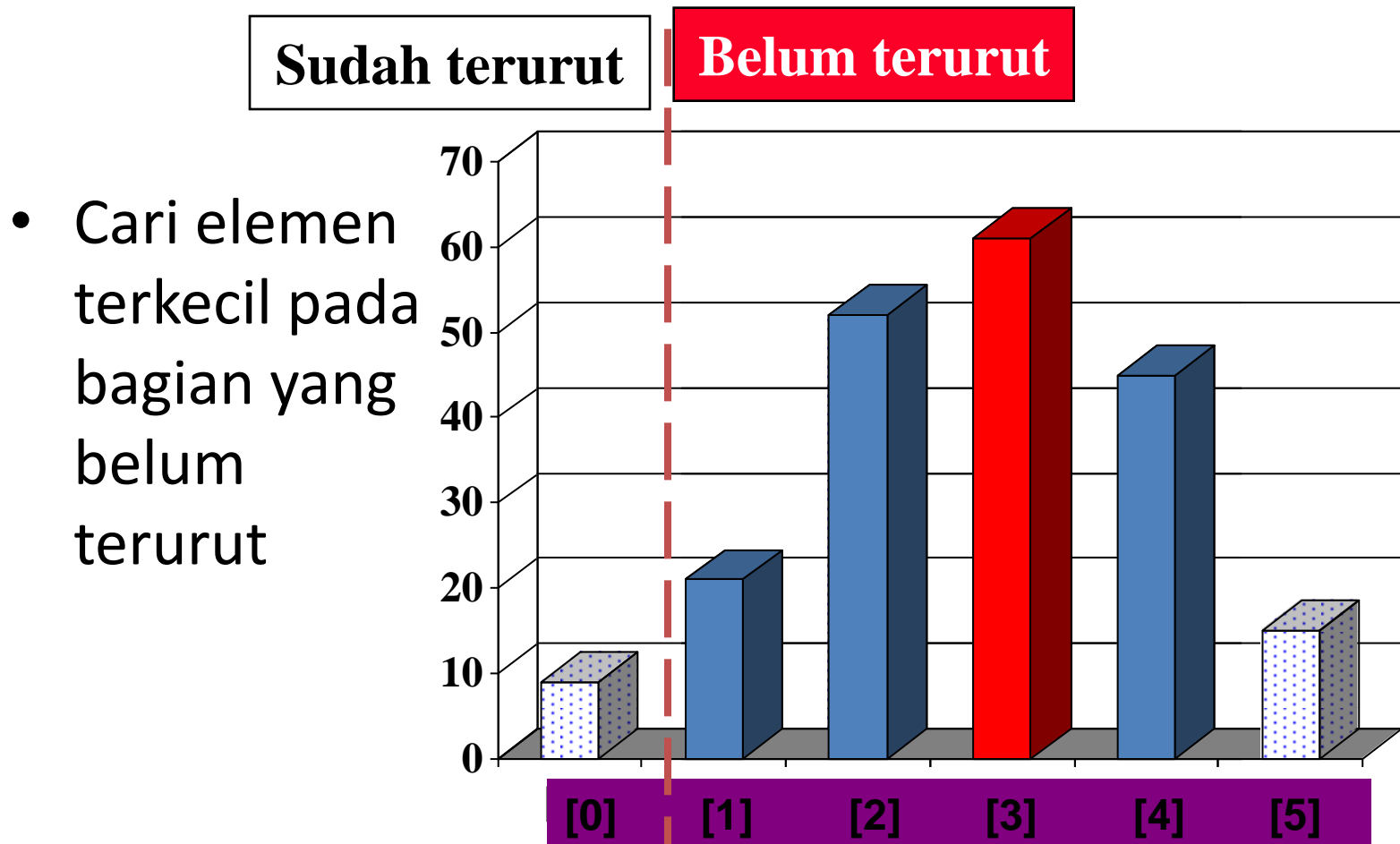


Selection Sort Algorithm – 4

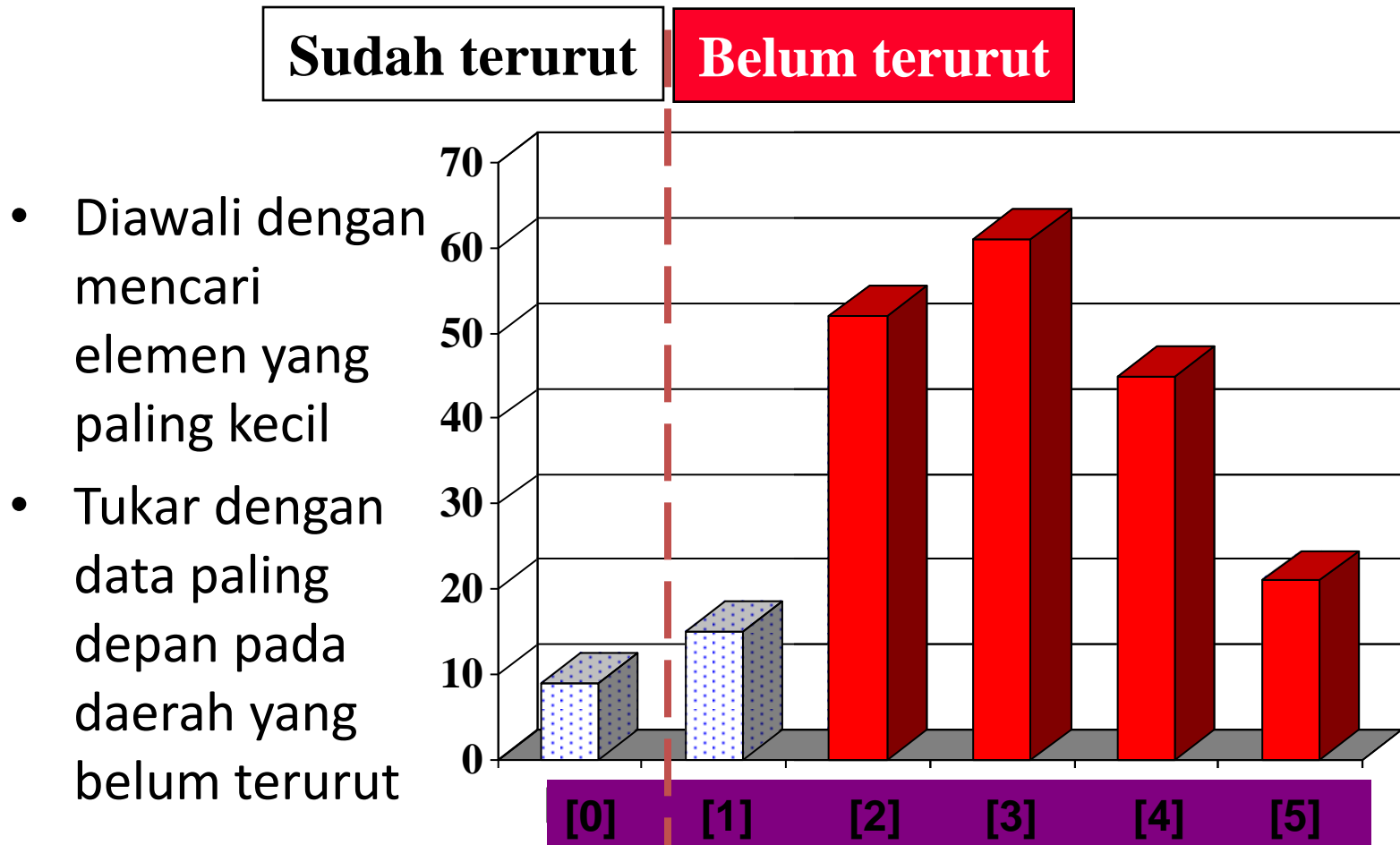
- Bagian dari Data yang sudah terurut



Selection Sort Algorithm – 5

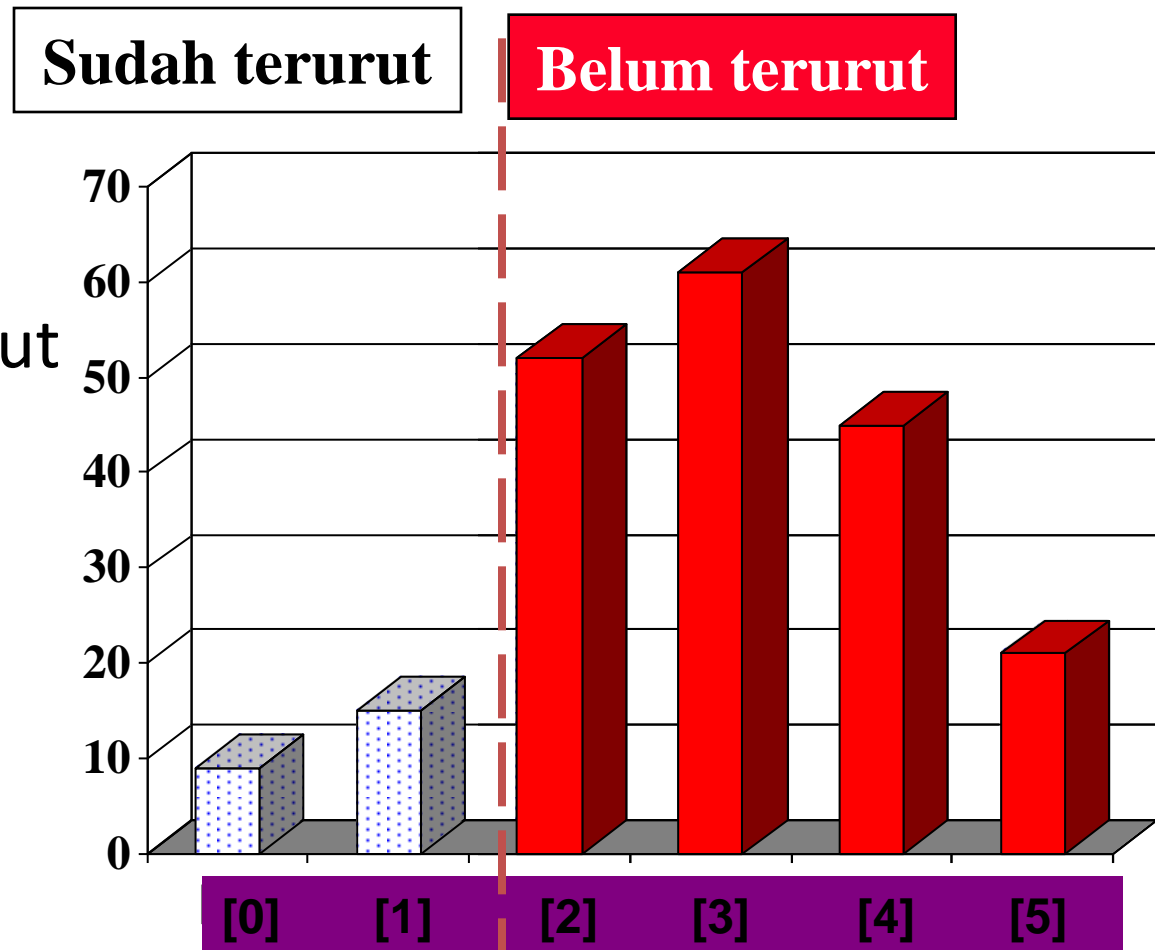


Selection Sort Algorithm – 6



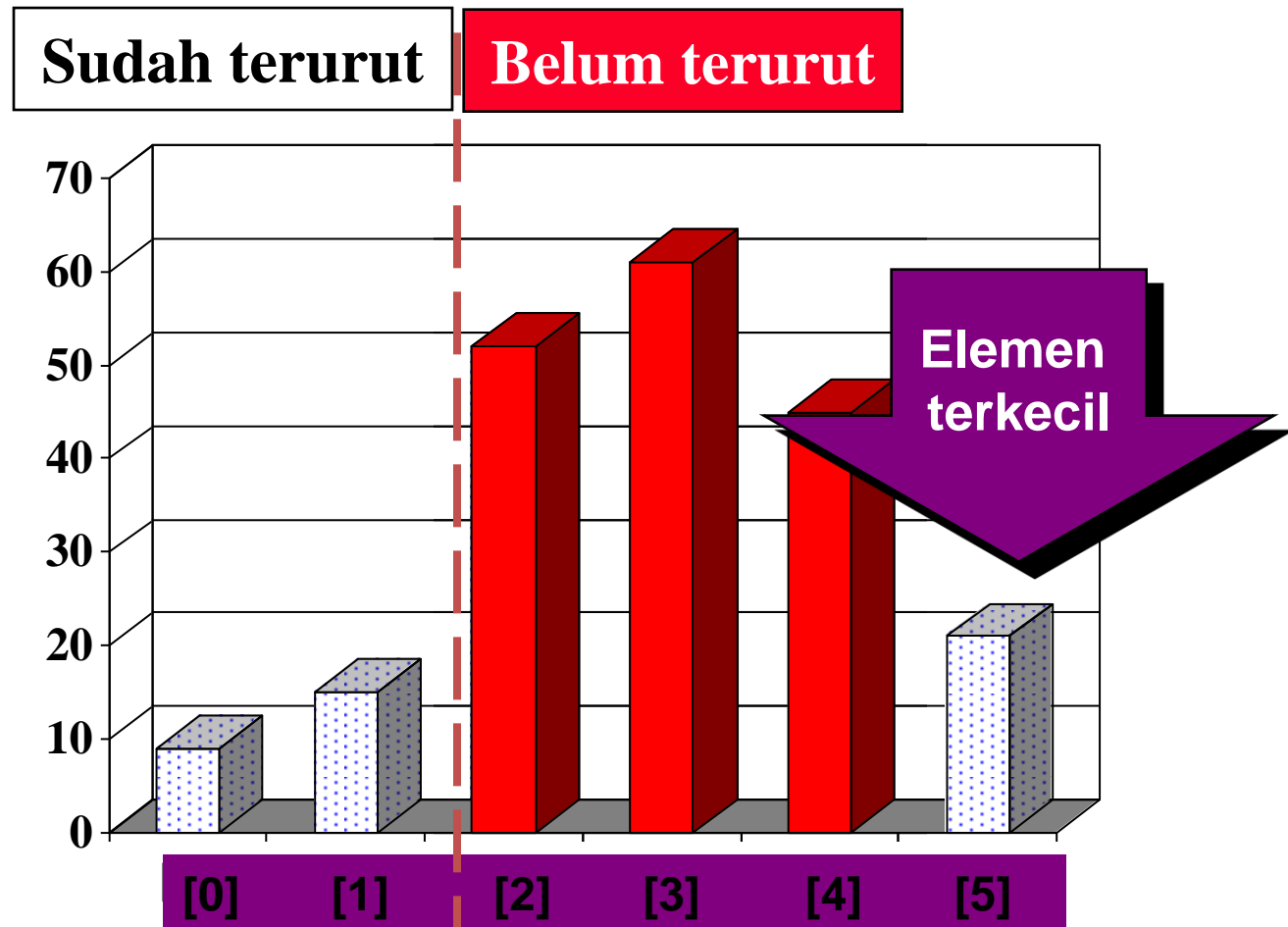
Selection Sort Algorithm – 7

- Dua data telah terurut



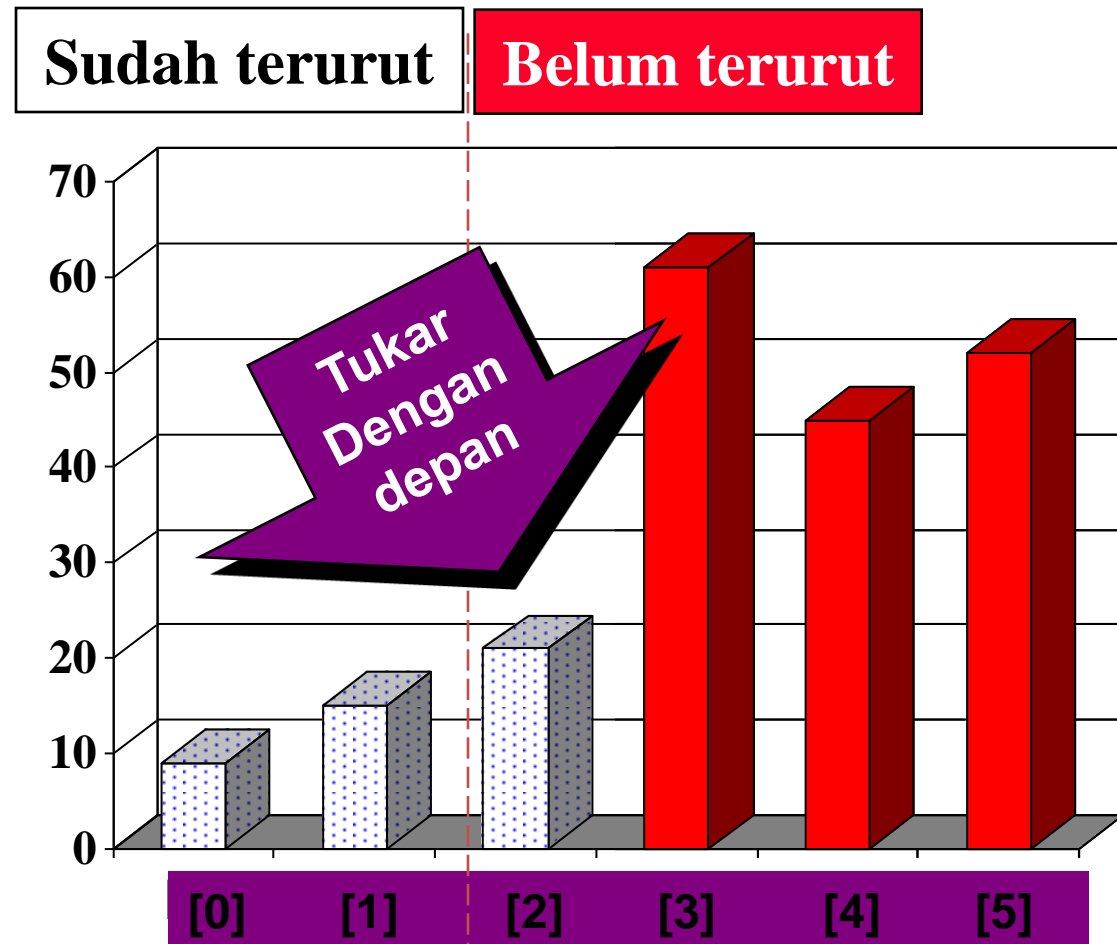
Selection Sort Algorithm – 8

- Proses dilanjutkan



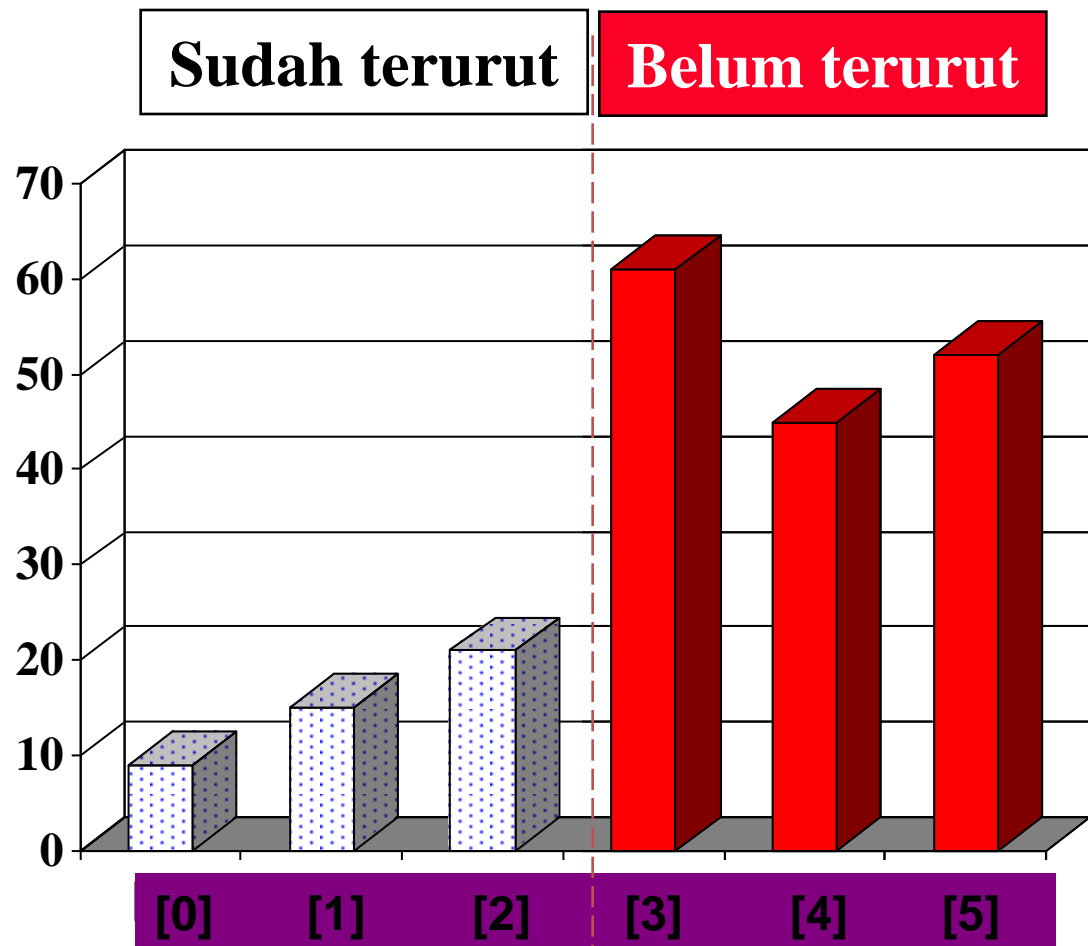
Selection Sort Algorithm – 9

- Proses dilanjutkan

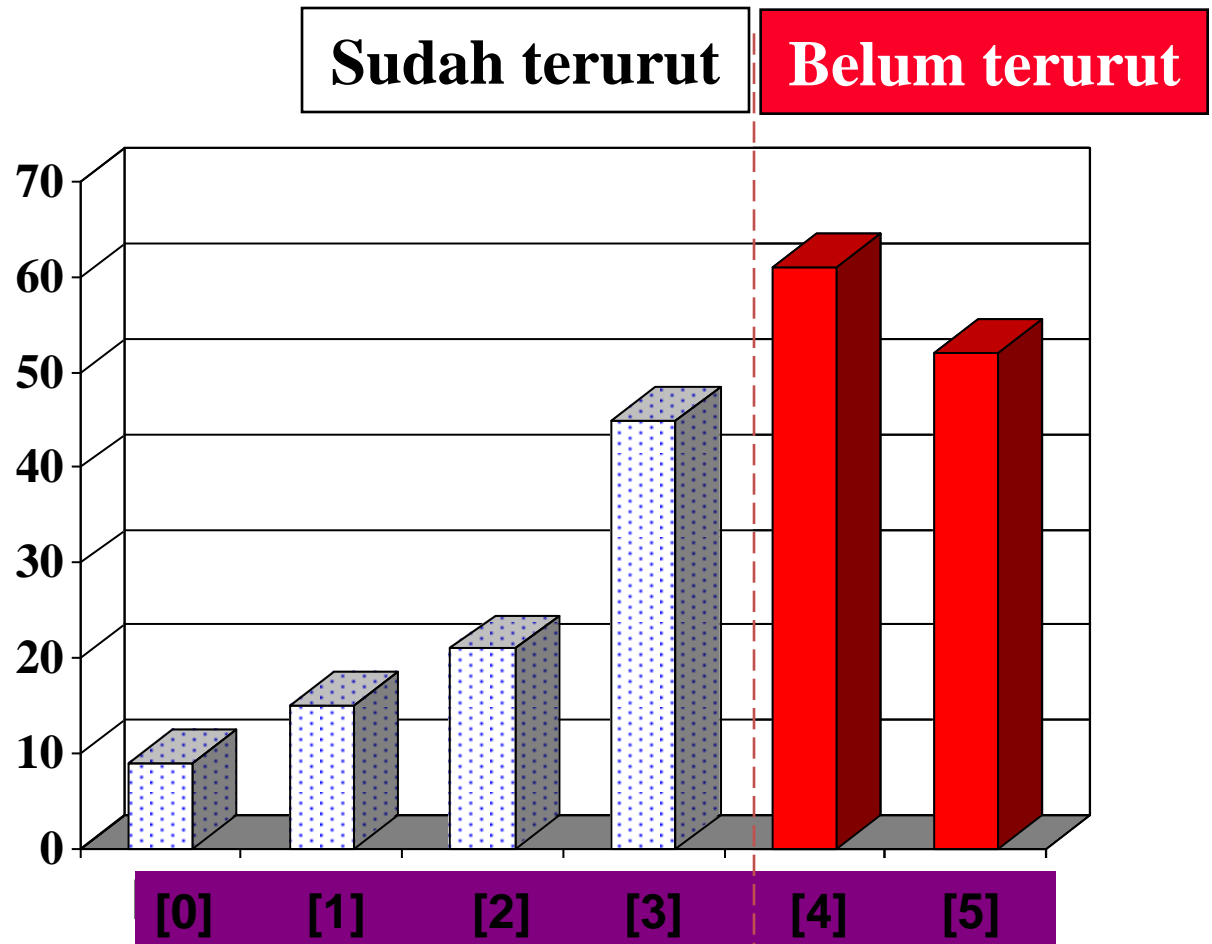


Selection Sort Algorithm – 10

- Proses dilanjutkan

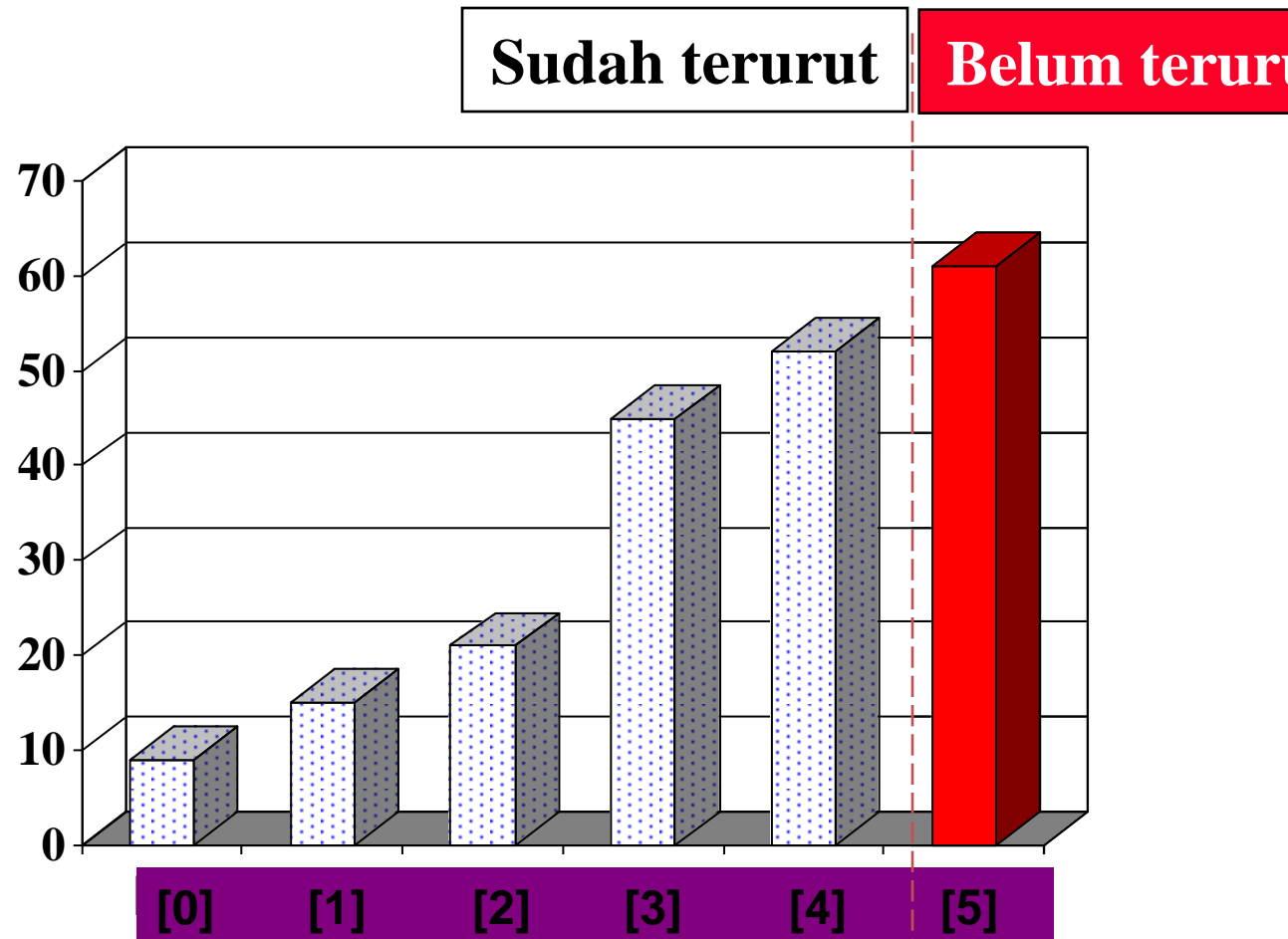


Selection Sort Algorithm – 11



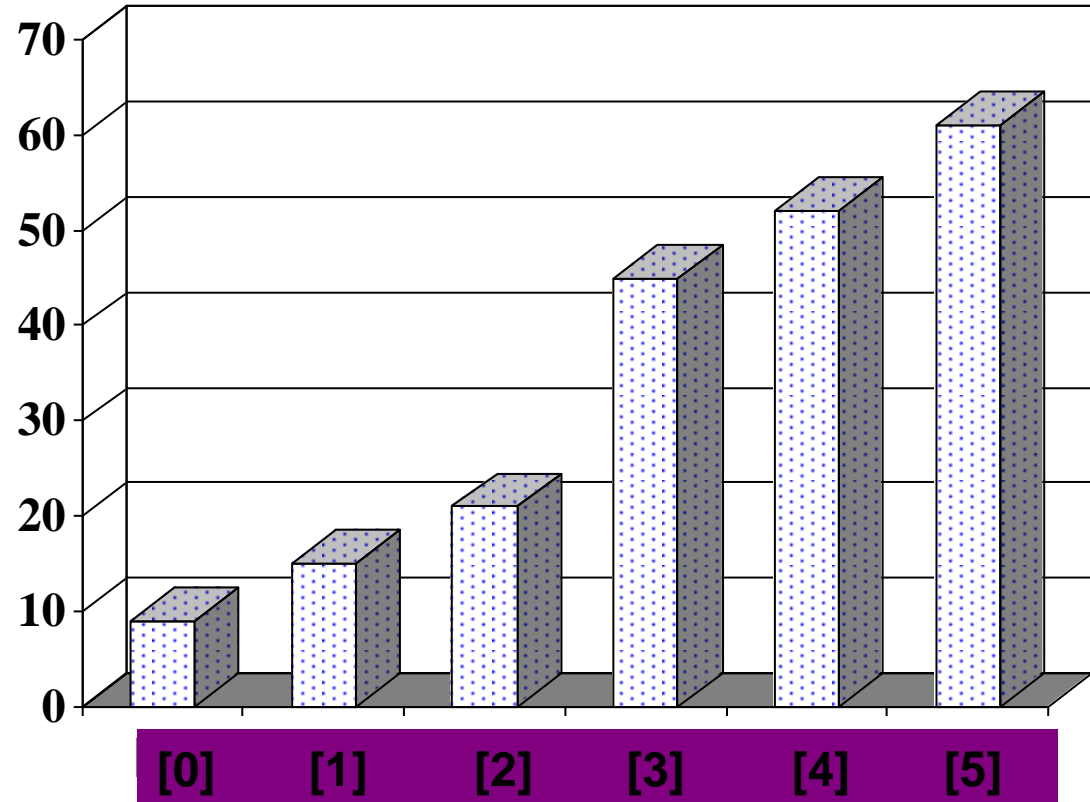
Selection Sort Algorithm – 12

- Berhenti pada saat bagian yang terurut tinggal hanya satu Data karena Data yang ada pasti bilangan yang paling besar



Selection Sort Algorithm – 13

- Data sudah terurut



Analisa Selection Sort

Basic Operasi ($A[j] < A[\text{min}]$)

Tidak ada Best Case dan Worst Case

Total pergeseran

$M = 3 * n - 1$ (pada setiap penukaran terjadi 3 x pergeseran)

Jumlah pembandingannya

$C = 1 + 2 + \dots + n - 1 = n * (n - 1) / 2$