



Bölüm 2: Sıralama Algoritmaları

Algoritmalar



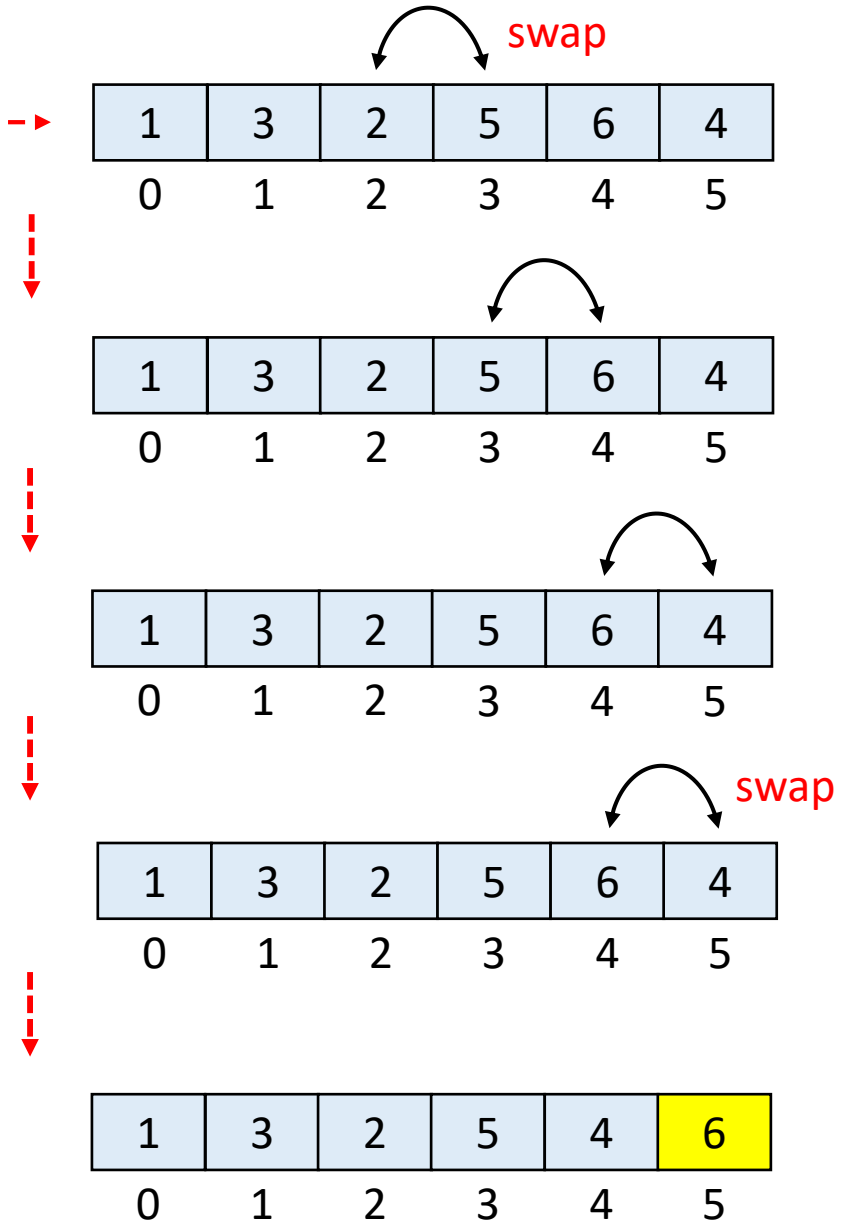
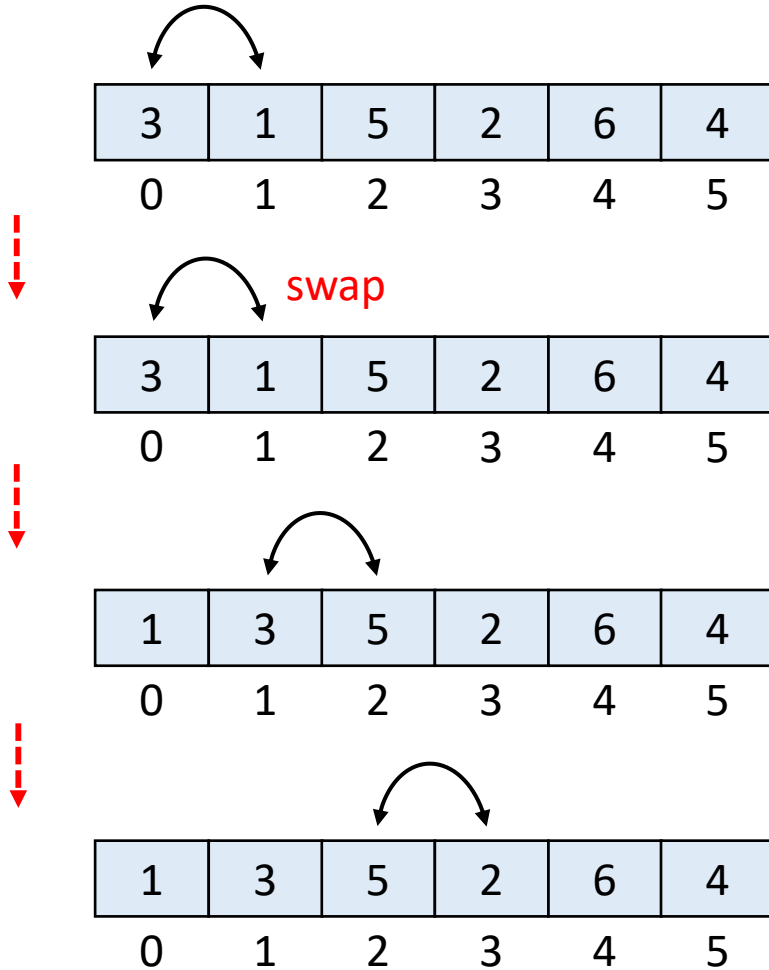
Kabarcık Sıralama (Bubble Sort)

- Her bir adımda dizideki büyük elemanlar dizinin sonuna doğru kaydırılır.
 - Komşu elemanlar ikili olarak birbiriyle karşılaştırılır.
 - Sonuca göre elemanların yerleri değiştirilir.

Kabarcık Sıralama



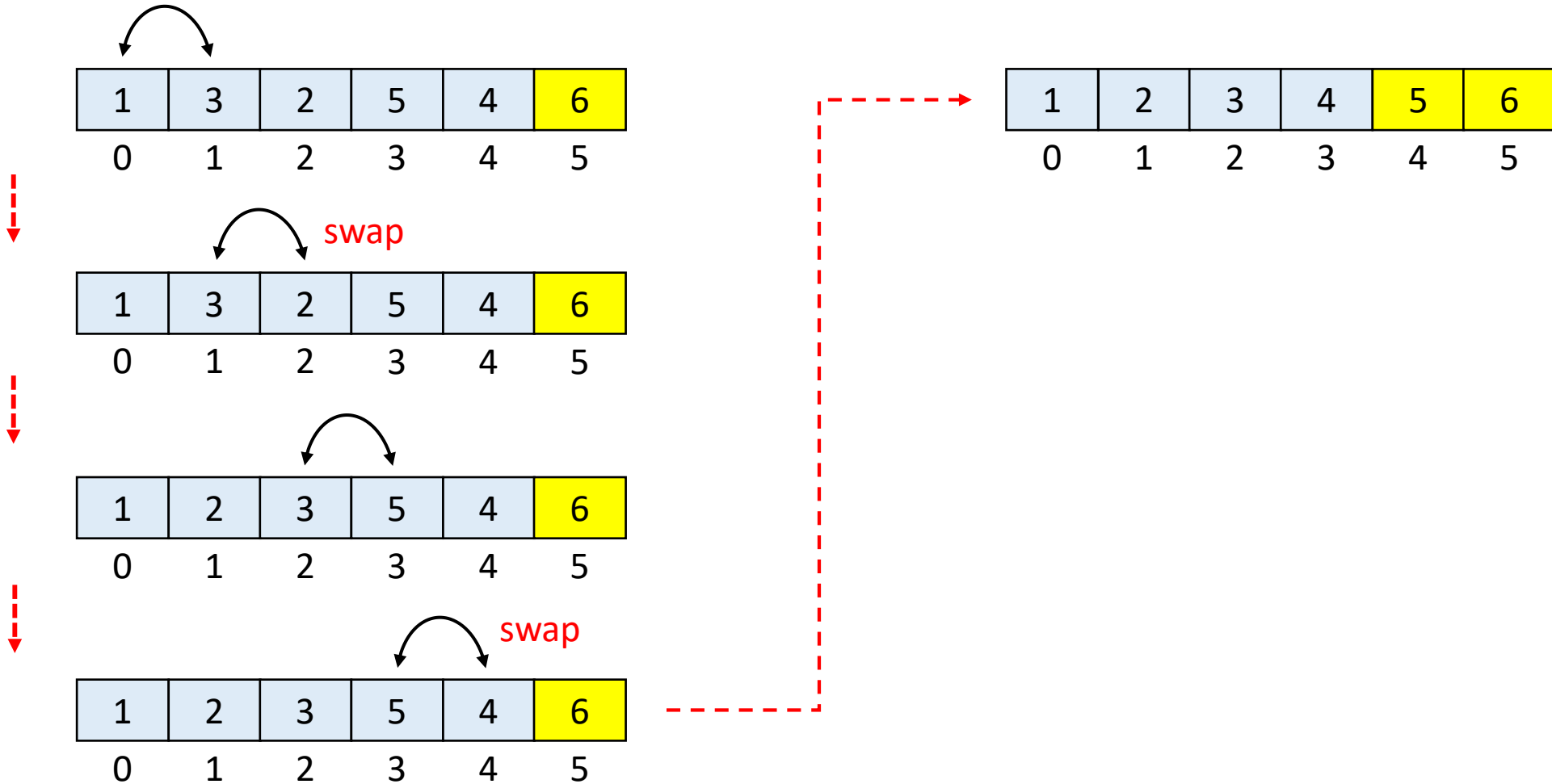
1. Tur



Kabarcık Sıralama



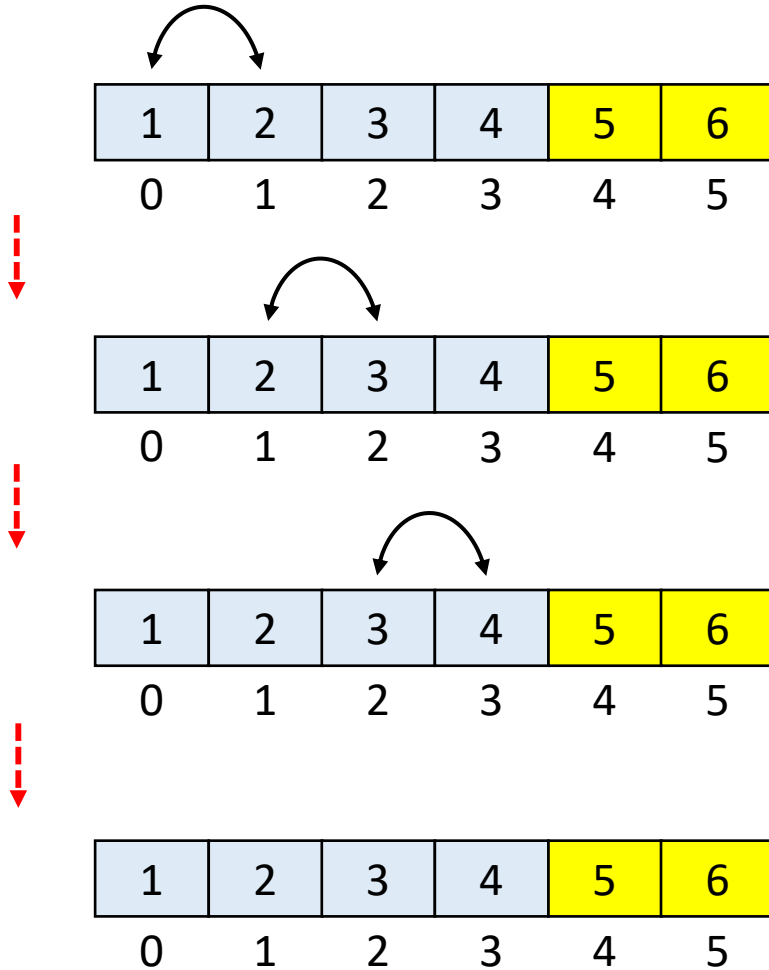
2. Tur



Kabarcık Sıralama



3. Tur



Kabarcık Sıralama



dizi[]	5	1	9	2	10
	0	1	2	3	4

```
→ boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]	5	1	9	2	10
	0	1	2	3	4

n = 5

```
→ boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]	5	1	9	2	10
	0	1	2	3	4

yerDegistiMi

n = 5

```
→ boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```


Kabarcık Sıralama



dizi[]	5	1	9	2	10
	0	1	2	3	4

yerDegistiMi
i = 0

n = 5

```
→ boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]	5	1	9	2	10
	0	1	2	3	4

yerDegistiMi = false
i = 0

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]	5	1	9	2	10
	0	1	2	3	4

yerDegistiMi = false

i = 0

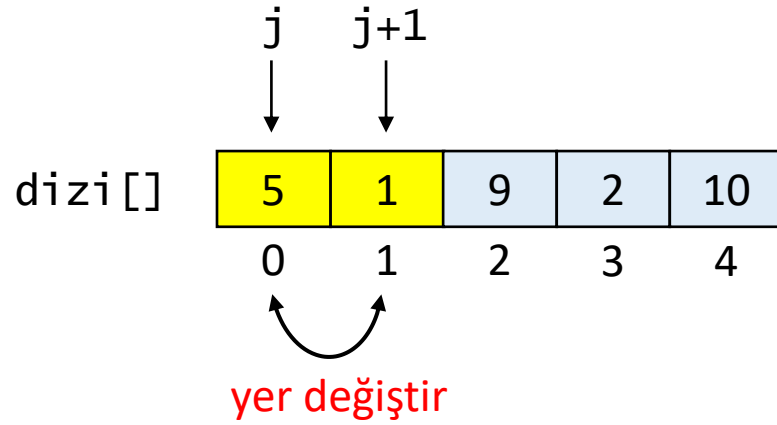
j = 0

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



yerDegistiMi = false

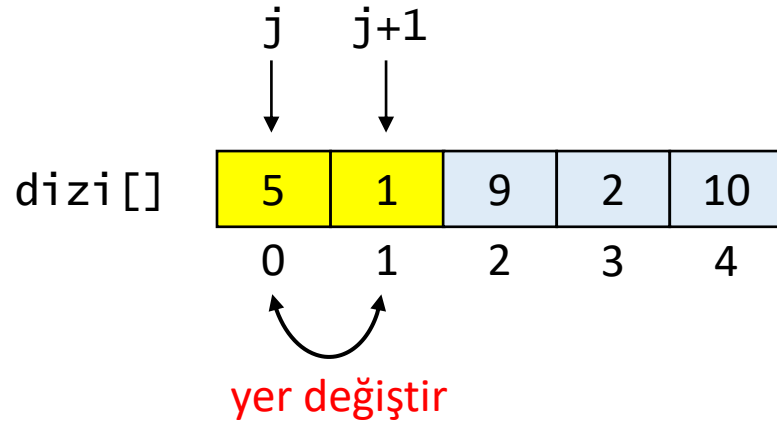
i = 0

j = 0

n = 5

```
boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
    yerDegistiMi = false;
    for(int j = 0; j < n - 1 - i; j++) {
        if(dizi[j] > dizi[j+1]) {
            int gecici = dizi[j];
            dizi[j] = dizi[j+1];
            dizi[j+1] = gecici;
            yerDegistiMi = true;
        }
    }
    if(yerDegistiMi == false) break;
}
```

Kabarcık Sıralama

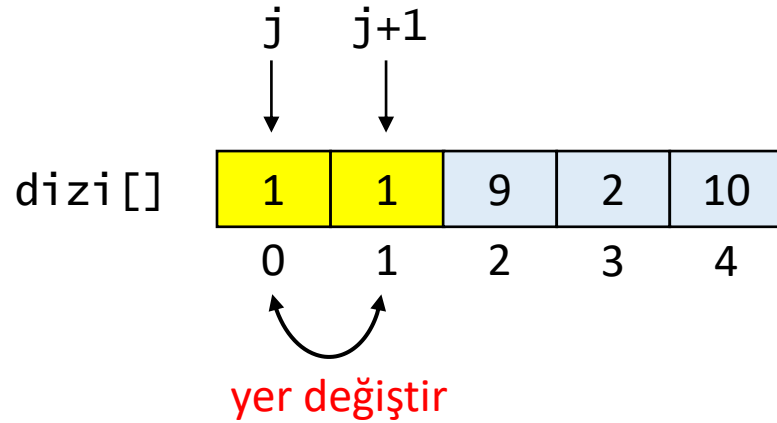


```
yerDegistiMi = false  
i = 0  
j = 0  
gecici = 5
```

$n = 5$

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama

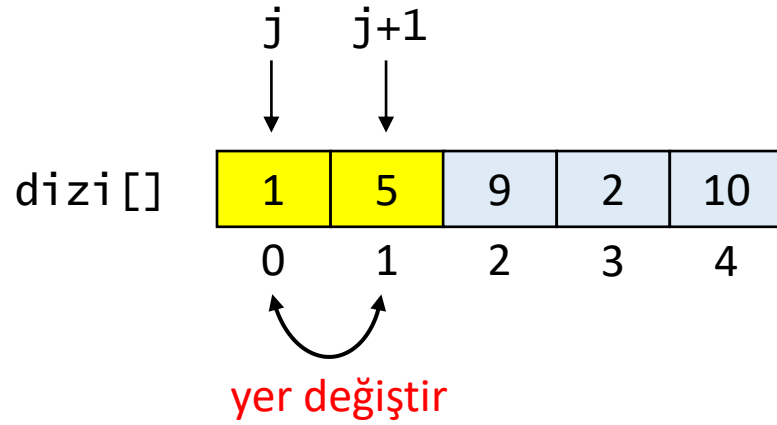


```
yerDegistiMi = false  
i = 0  
j = 0  
gecici = 5
```

$n = 5$

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama

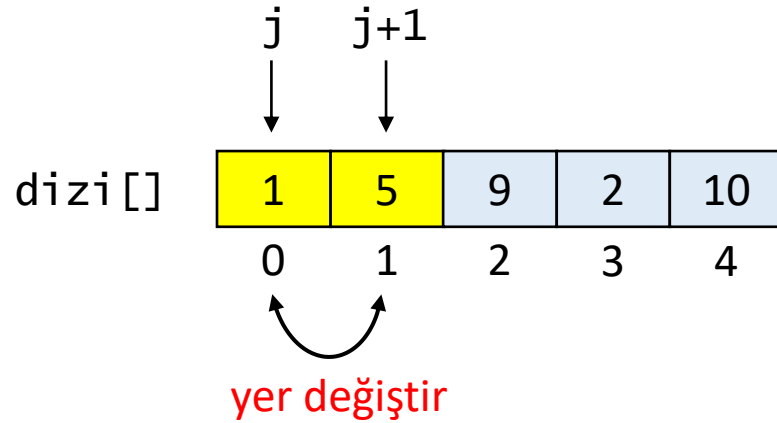


```
yerDegistiMi = false  
i = 0  
j = 0  
gecici = 5
```

$n = 5$

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama

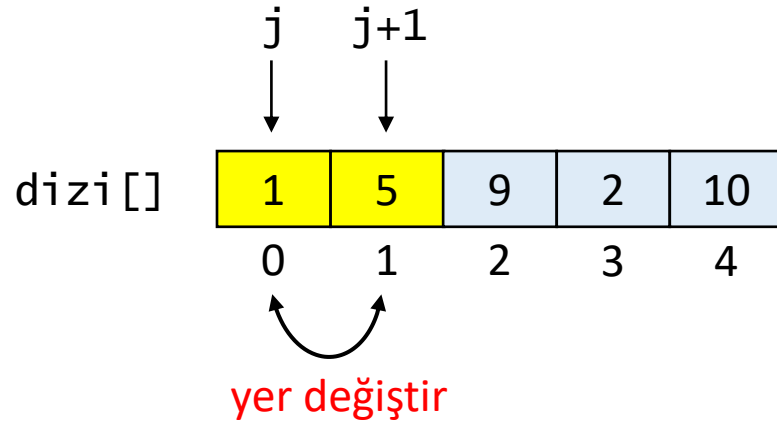


```
yerDegistiMi = true  
i = 0  
j = 0  
gecici = 5
```

$n = 5$

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```


Kabarcık Sıralama



```
yerDegistiMi = true  
i = 0  
j = 0  
gecici = 5
```

$n = 5$

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]

1	5	9	2	10
0	1	2	3	4

yerDegistiMi = true

i = 0

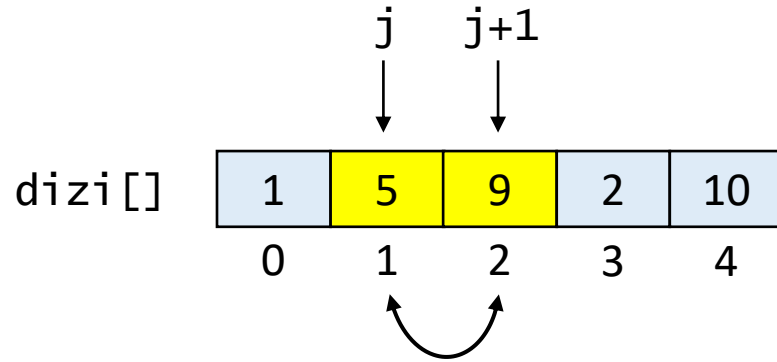
j = 1

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



`yerDegistiMi = true`

`i = 0`

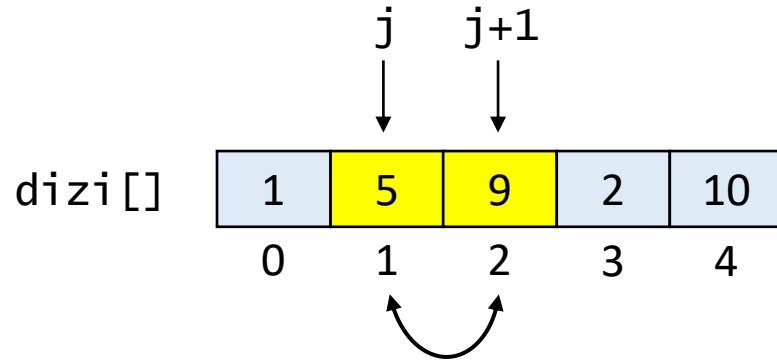
`j = 1`

`n = 5`



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



`yerDegistiMi = true`

`i = 0`

`j = 1`

`n = 5`

```
boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
    yerDegistiMi = false;
    for(int j = 0; j < n - 1 - i; j++) {
        if(dizi[j] > dizi[j+1]) {
            int gecici = dizi[j];
            dizi[j] = dizi[j+1];
            dizi[j+1] = gecici;
            yerDegistiMi = true;
        }
    }
    if(yerDegistiMi == false) break;
}
```

Kabarcık Sıralama



dizi[]	1	5	9	2	10
	0	1	2	3	4

yerDegistiMi = true

i = 0

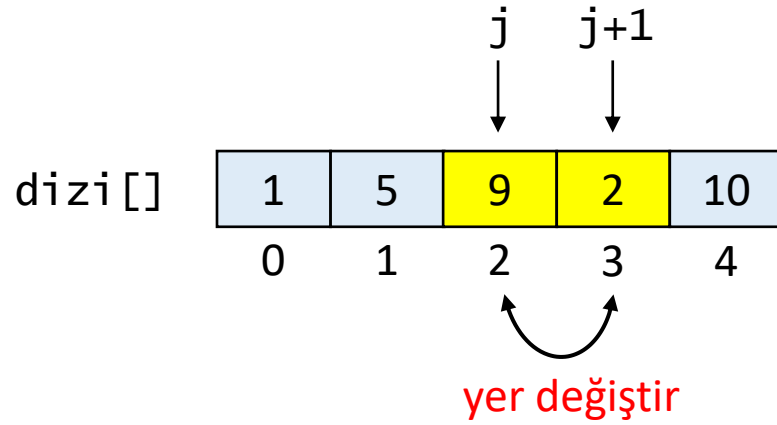
j = 2

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



`yerDegistiMi = true`

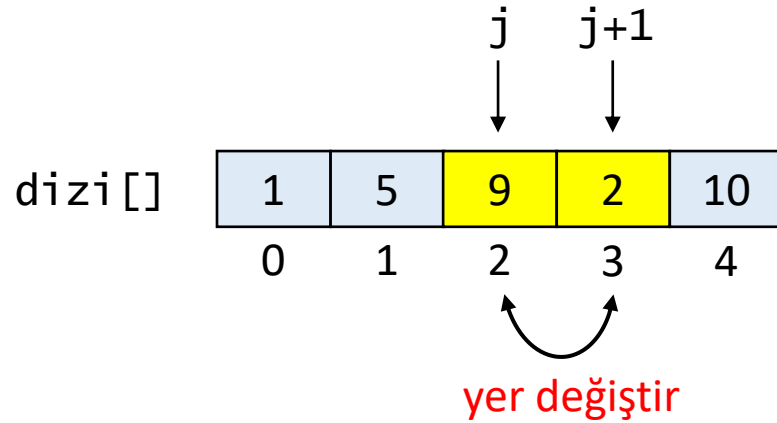
`i = 0`

`j = 2`

`n = 5`

```
boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
    yerDegistiMi = false;
    for(int j = 0; j < n - 1 - i; j++) {
        if(dizi[j] > dizi[j+1]) {
            int gecici = dizi[j];
            dizi[j] = dizi[j+1];
            dizi[j+1] = gecici;
            yerDegistiMi = true;
        }
    }
    if(yerDegistiMi == false) break;
}
```

Kabarcık Sıralama

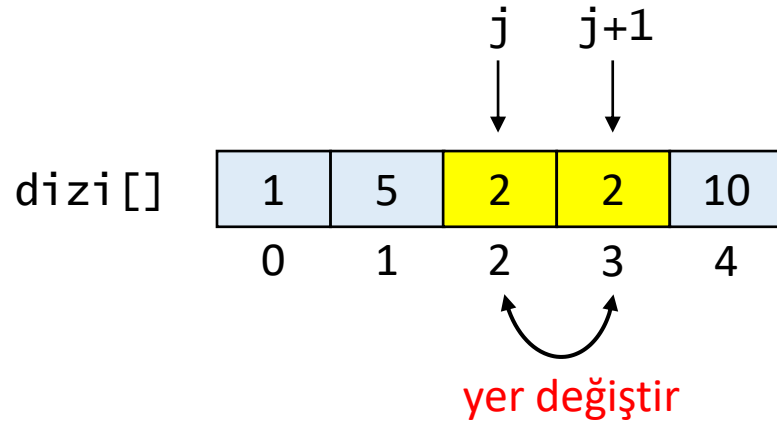


```
yerDegistiMi = true  
i = 0  
j = 2  
gecici = 9
```

```
n = 5
```

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama

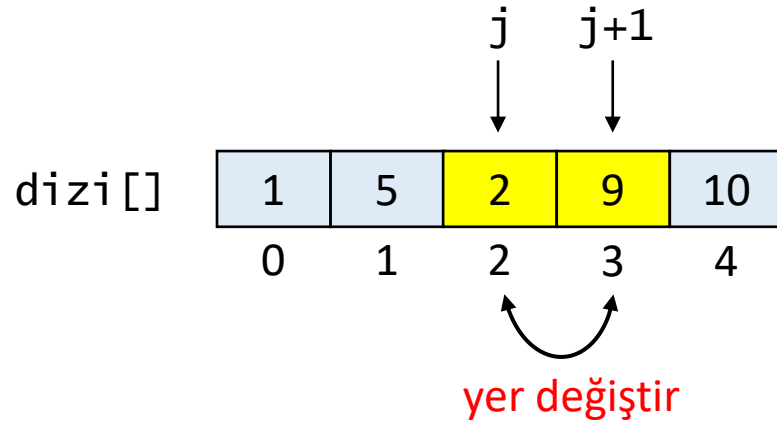


```
yerDegistiMi = true  
i = 0  
j = 2  
gecici = 9
```

```
n = 5
```

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```


Kabarcık Sıralama

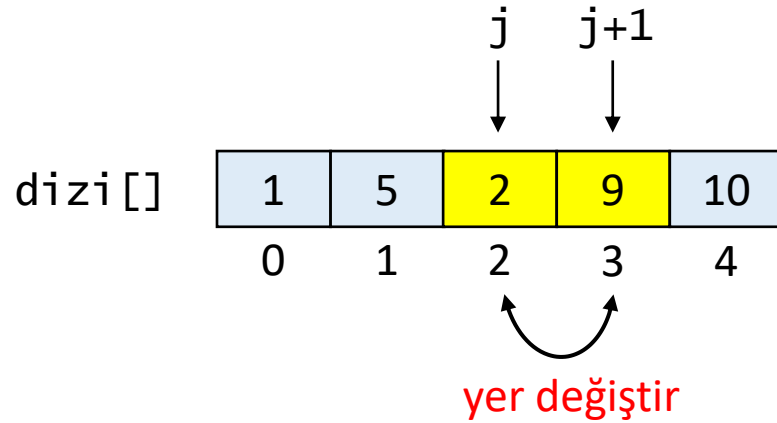


```
yerDegistiMi = true  
i = 0  
j = 2  
gecici = 9
```

```
n = 5
```

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama

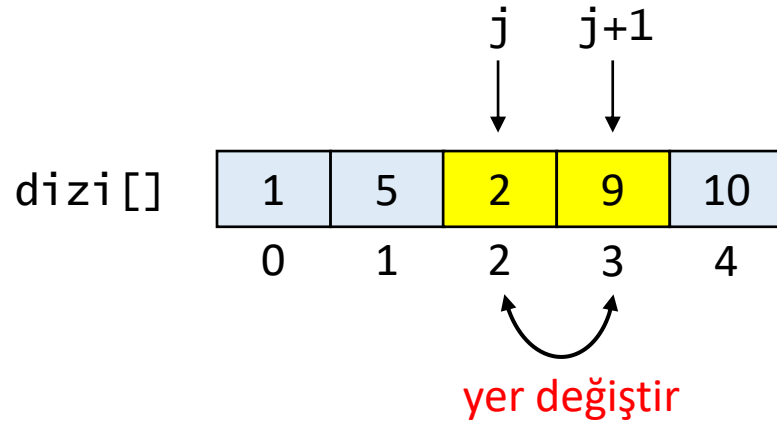


```
yerDegistiMi = true  
i = 0  
j = 2  
gecici = 9
```

```
n = 5
```

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



```
yerDegistiMi = true  
i = 0  
j = 2  
gecici = 9
```

```
n = 5
```

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]	1	5	2	9	10
	0	1	2	3	4

yerDegistiMi = true

i = 0

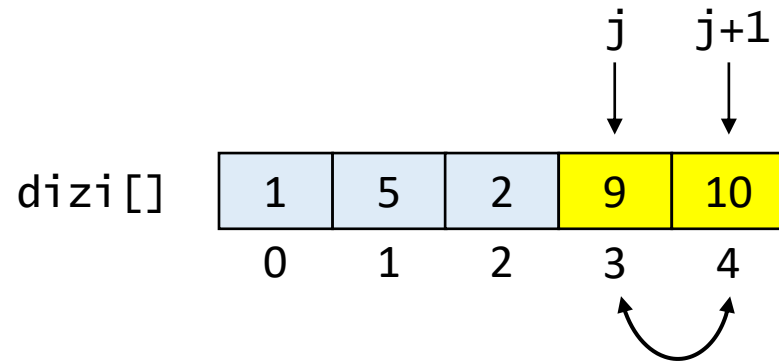
j = 3

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



`yerDegistiMi = true`

`i = 0`

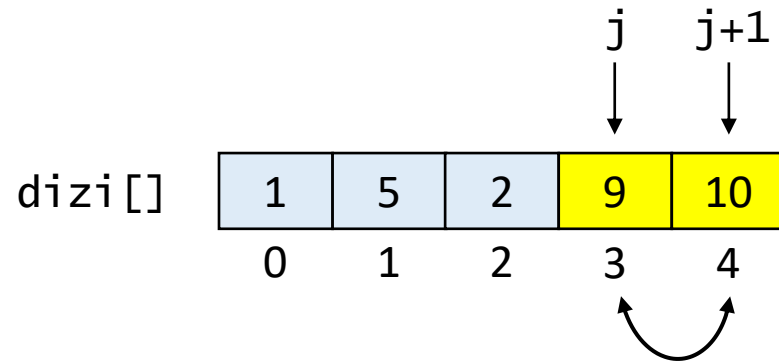
`j = 3`

`n = 5`



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



`yerDegistiMi = true`

`i = 0`

`j = 3`

`n = 5`

```
boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
    yerDegistiMi = false;
    for(int j = 0; j < n - 1 - i; j++) {
        if(dizi[j] > dizi[j+1]) {
            int gecici = dizi[j];
            dizi[j] = dizi[j+1];
            dizi[j+1] = gecici;
            yerDegistiMi = true;
        }
    }
    if(yerDegistiMi == false) break;
}
```

Kabarcık Sıralama



dizi[]	1	5	2	9	10
	0	1	2	3	4

yerDegistiMi = true

i = 0

j = 4

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]	1	5	2	9	10
	0	1	2	3	4

yerDegistiMi = true
i = 0

n = 5

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```


Kabarcık Sıralama



dizi[]	1	5	2	9	10
	0	1	2	3	4

yerDegistiMi = true
i = 1

n = 5

```
→ boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]	1	5	2	9	10
	0	1	2	3	4

yerDegistiMi = false
i = 1

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]

1	5	2	9	10
0	1	2	3	4

yerDegistiMi = false

i = 1

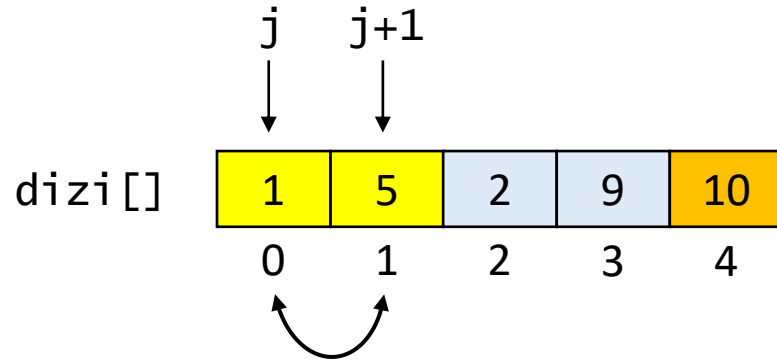
j = 0

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



yerDegistiMi = false

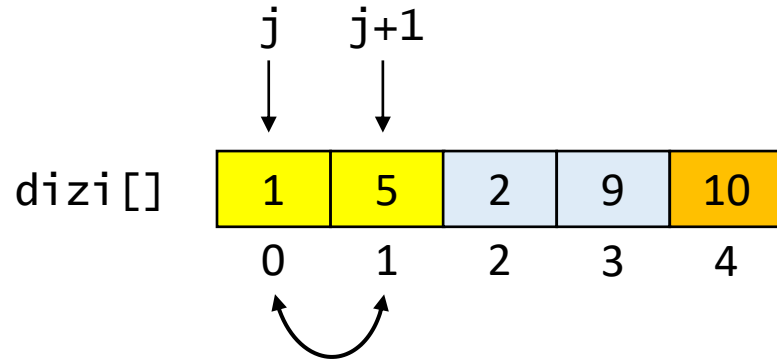
i = 1

j = 0

n = 5

```
boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
    yerDegistiMi = false;
    for(int j = 0; j < n - 1 - i; j++) {
        if(dizi[j] > dizi[j+1]) {
            int gecici = dizi[j];
            dizi[j] = dizi[j+1];
            dizi[j+1] = gecici;
            yerDegistiMi = true;
        }
    }
    if(yerDegistiMi == false) break;
}
```

Kabarcık Sıralama



yerDegistiMi = false

i = 1

j = 0

n = 5

```
boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
    yerDegistiMi = false;
    for(int j = 0; j < n - 1 - i; j++) {
        if(dizi[j] > dizi[j+1]) {
            int gecici = dizi[j];
            dizi[j] = dizi[j+1];
            dizi[j+1] = gecici;
            yerDegistiMi = true;
        }
    }
    if(yerDegistiMi == false) break;
}
```

Kabarcık Sıralama



dizi[]	1	5	2	9	10
	0	1	2	3	4

yerDegistiMi = false

i = 1

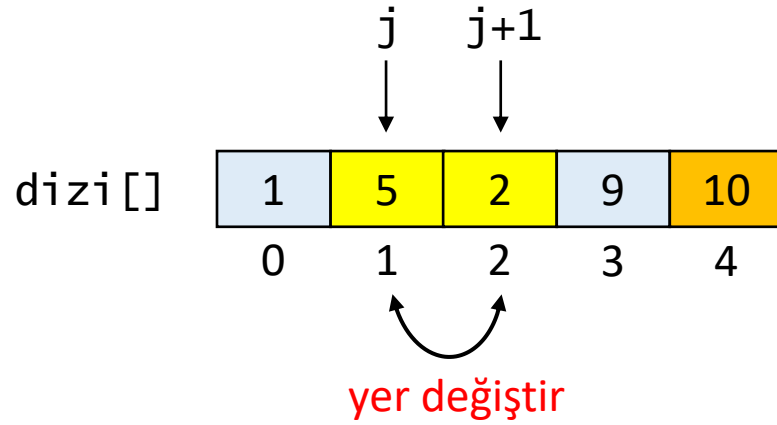
j = 1

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



`yerDegistiMi = false`

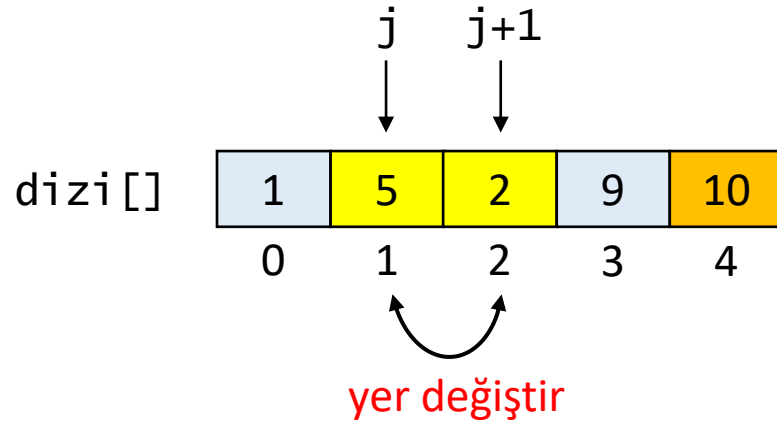
`i = 1`

`j = 1`

`n = 5`

```
boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
    yerDegistiMi = false;
    for(int j = 0; j < n - 1 - i; j++) {
        if(dizi[j] > dizi[j+1]) {
            int gecici = dizi[j];
            dizi[j] = dizi[j+1];
            dizi[j+1] = gecici;
            yerDegistiMi = true;
        }
    }
    if(yerDegistiMi == false) break;
}
```

Kabarcık Sıralama

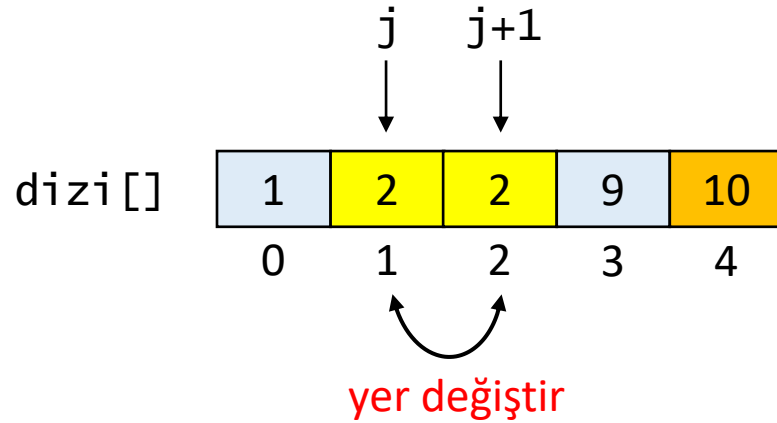


```
yerDegistiMi = false  
i = 1  
j = 1  
gecici = 5
```

```
n = 5
```

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```


Kabarcık Sıralama

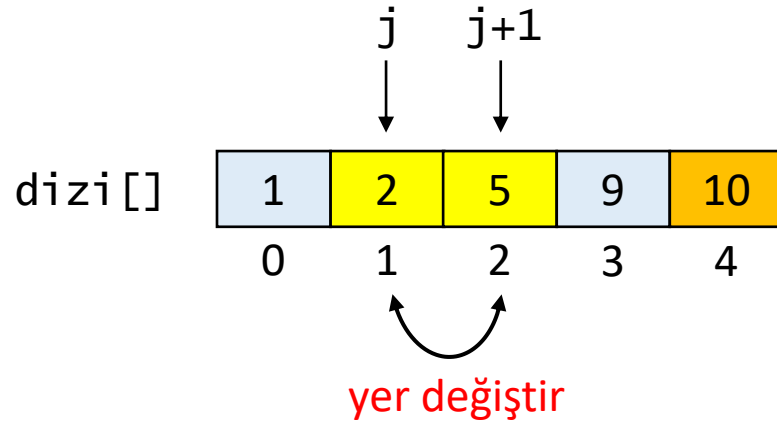


```
yerDegistiMi = false  
i = 1  
j = 1  
gecici = 5
```

`n = 5`

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama

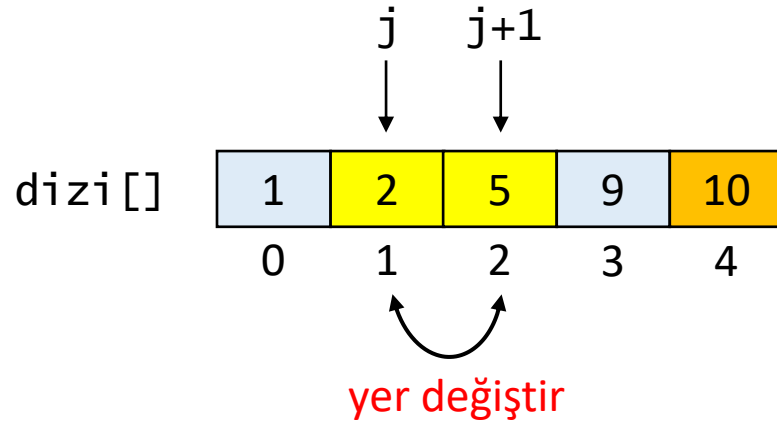


```
yerDegistiMi = false  
i = 1  
j = 1  
gecici = 5
```

`n = 5`

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama

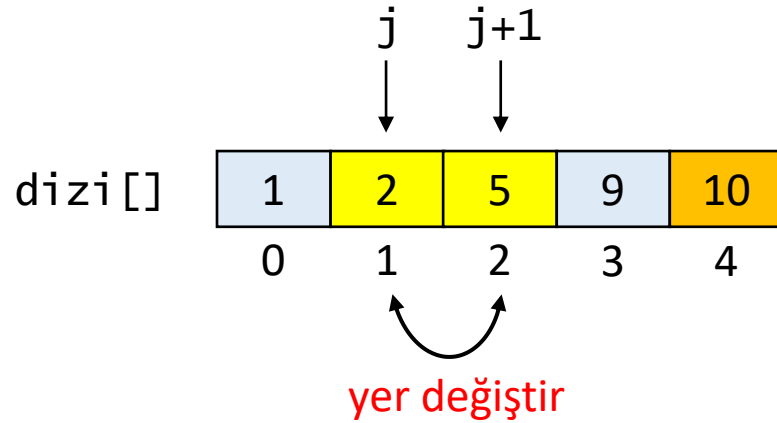


```
yerDegistiMi = true  
i = 1  
j = 1  
gecici = 5
```

```
n = 5
```

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



```
yerDegistiMi = true  
i = 1  
j = 1  
gecici = 5
```

`n = 5`

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]

1	2	5	9	10
0	1	2	3	4

yerDegistiMi = true

i = 1

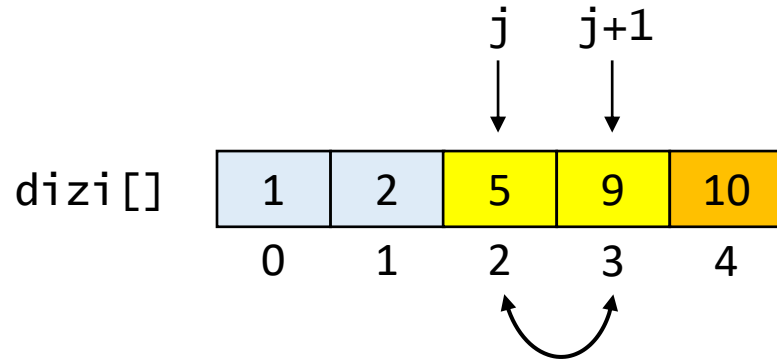
j = 2

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



`yerDegistiMi = true`

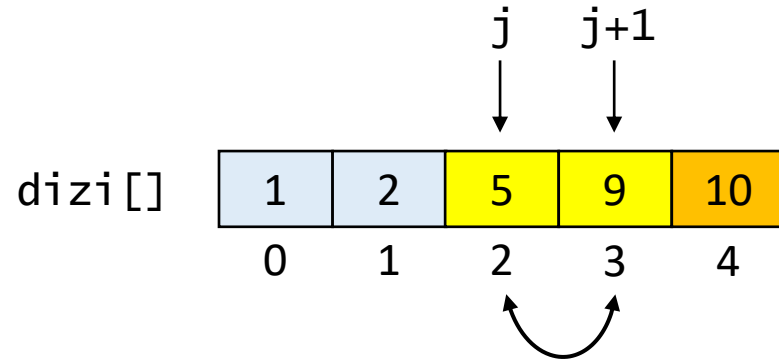
`i = 1`

`j = 2`

`n = 5`

```
boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
    yerDegistiMi = false;
    for(int j = 0; j < n - 1 - i; j++) {
        if(dizi[j] > dizi[j+1]) {
            int gecici = dizi[j];
            dizi[j] = dizi[j+1];
            dizi[j+1] = gecici;
            yerDegistiMi = true;
        }
    }
    if(yerDegistiMi == false) break;
}
```

Kabarcık Sıralama



`yerDegistiMi = true`

`i = 1`

`j = 2`

`n = 5`

```
boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
    yerDegistiMi = false;
    for(int j = 0; j < n - 1 - i; j++) {
        if(dizi[j] > dizi[j+1]) {
            int gecici = dizi[j];
            dizi[j] = dizi[j+1];
            dizi[j+1] = gecici;
            yerDegistiMi = true;
        }
    }
    if(yerDegistiMi == false) break;
}
```

Kabarcık Sıralama



dizi[]	1	2	5	9	10
	0	1	2	3	4

yerDegistiMi = true

i = 1

j = 3

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```


Kabarcık Sıralama



dizi[]	1	2	5	9	10
	0	1	2	3	4

yerDegistiMi = true
i = 1

n = 5

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]	1	2	5	9	10
	0	1	2	3	4

yerDegistiMi = true
i = 2

n = 5

```
→ boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]	1	2	5	9	10
	0	1	2	3	4

yerDegistiMi = false
i = 2

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]	1	2	5	9	10
	0	1	2	3	4

yerDegistiMi = false

i = 2

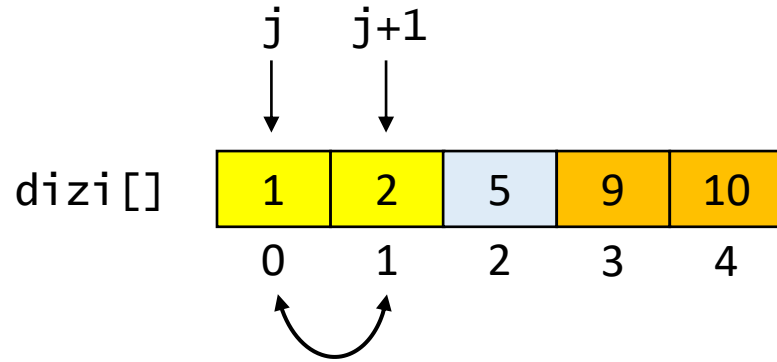
j = 0

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



yerDegistiMi = false

i = 2

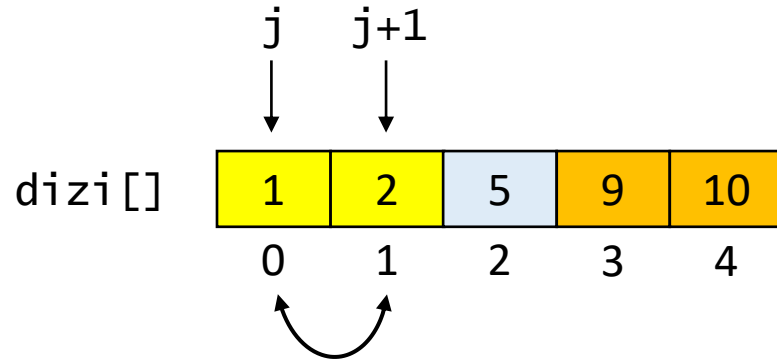
j = 0

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



yerDegistiMi = false

i = 2

j = 0

n = 5

```
boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
    yerDegistiMi = false;
    for(int j = 0; j < n - 1 - i; j++) {
        if(dizi[j] > dizi[j+1]) {
            int gecici = dizi[j];
            dizi[j] = dizi[j+1];
            dizi[j+1] = gecici;
            yerDegistiMi = true;
        }
    }
    if(yerDegistiMi == false) break;
}
```

Kabarcık Sıralama



dizi[]	1	2	5	9	10
	0	1	2	3	4

yerDegistiMi = false

i = 2

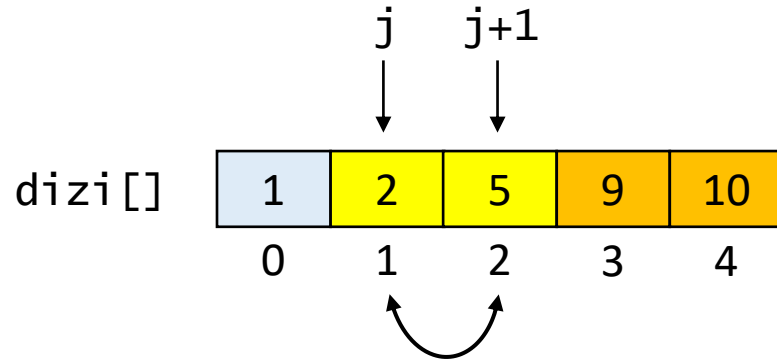
j = 1

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



`yerDegistiMi = false`

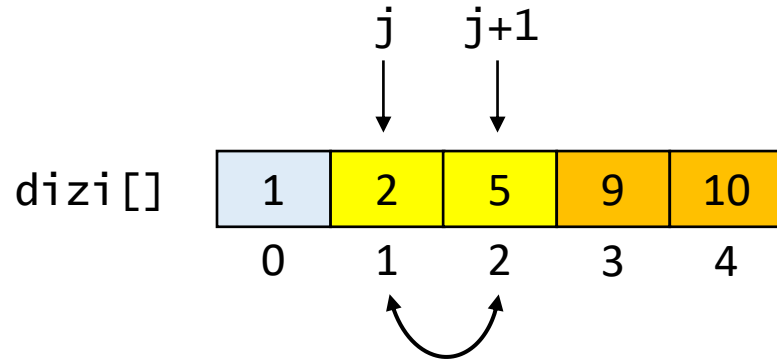
`i = 2`

`j = 1`

`n = 5`

```
boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
    yerDegistiMi = false;
    for(int j = 0; j < n - 1 - i; j++) {
        if(dizi[j] > dizi[j+1]) {
            int gecici = dizi[j];
            dizi[j] = dizi[j+1];
            dizi[j+1] = gecici;
            yerDegistiMi = true;
        }
    }
    if(yerDegistiMi == false) break;
}
```


Kabarcık Sıralama



`yerDegistiMi = false`

`i = 2`

`j = 1`

`n = 5`

```
boolean yerDegistiMi;
for(int i = 0; i < n - 1; i++) {
    yerDegistiMi = false;
    for(int j = 0; j < n - 1 - i; j++) {
        if(dizi[j] > dizi[j+1]) {
            int gecici = dizi[j];
            dizi[j] = dizi[j+1];
            dizi[j+1] = gecici;
            yerDegistiMi = true;
        }
    }
    if(yerDegistiMi == false) break;
}
```

Kabarcık Sıralama



dizi[]	1	2	5	9	10
	0	1	2	3	4

yerDegistiMi = false

i = 2

j = 2

n = 5



```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]

1	2	5	9	10
0	1	2	3	4

yerDegistiMi = false
i = 2

n = 5

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```

Kabarcık Sıralama



dizi[]

1	2	5	9	10
0	1	2	3	4

```
boolean yerDegistiMi;  
for(int i = 0; i < n - 1; i++) {  
    yerDegistiMi = false;  
    for(int j = 0; j < n - 1 - i; j++) {  
        if(dizi[j] > dizi[j+1]) {  
            int gecici = dizi[j];  
            dizi[j] = dizi[j+1];  
            dizi[j+1] = gecici;  
            yerDegistiMi = true;  
        }  
    }  
    if(yerDegistiMi == false) break;  
}
```



Eklemeli Sıralama (Insertion Sort)

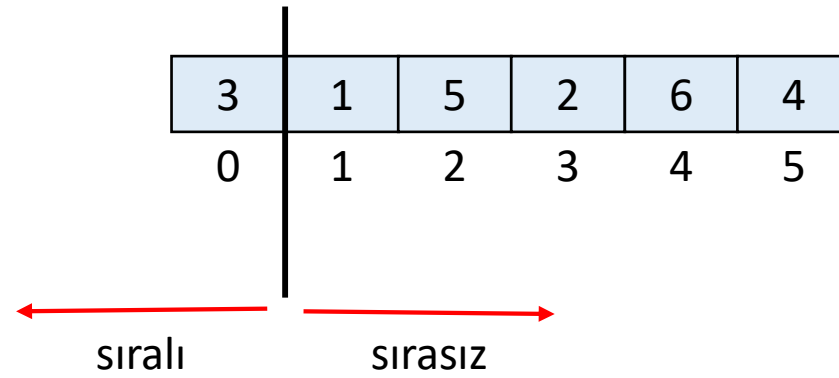
- Eldeki oyun kartlarını sıralamaya benzer.
- Verilen dizi iki parçaya ayrılır:
 - Sıralı kesim
 - Sırasız kesim
- Sırasız kısımdan ilk eleman, sıralı kısımda doğru konuma yerleştirilir.
- Sırasız kısımda bulunan diğer elemanların kaydırılmasını gerektirir.

Eklemeli Sıralama



3	1	5	2	6	4
0	1	2	3	4	5

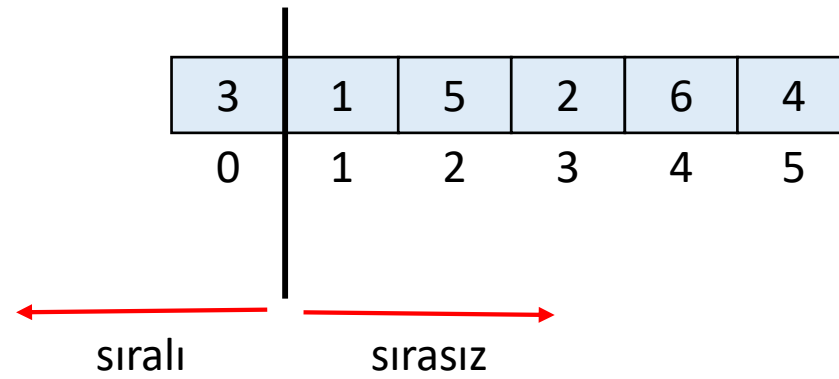
Eklemeli Sıralama



Eklemeli Sıralama



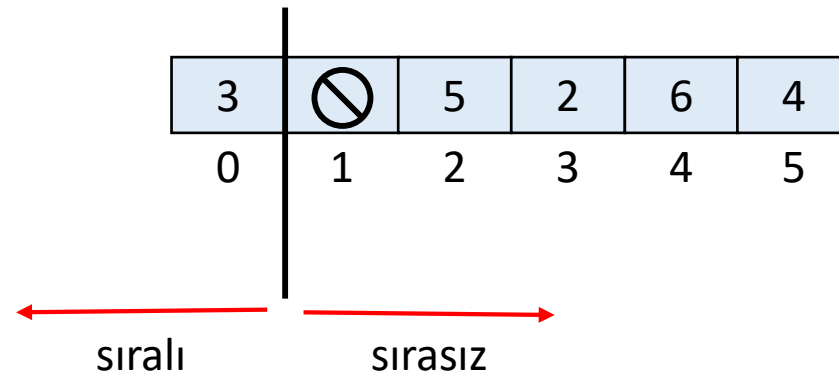
gecici = 1



Eklemeli Sıralama



gecici = 1

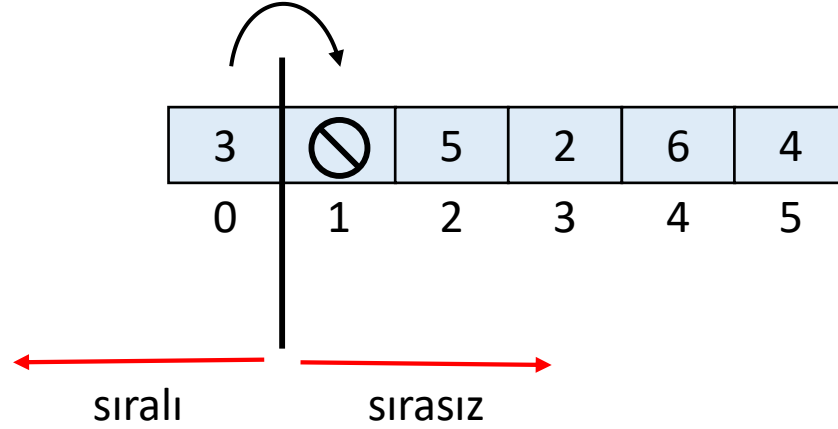


Eklemeli Sıralama



gecici = 1

kaydır

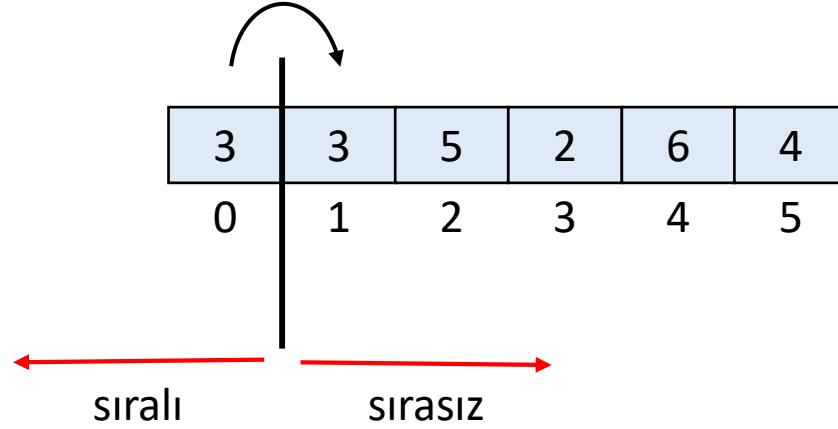


Eklemeli Sıralama



gecici = 1

kaydır

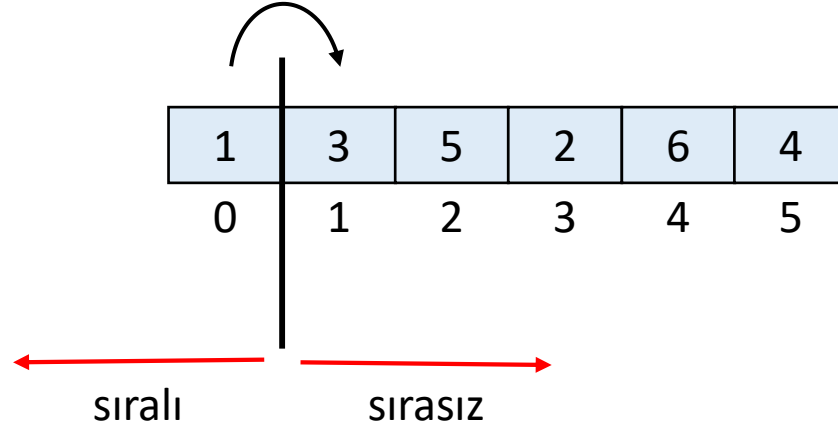


Eklemeli Sıralama

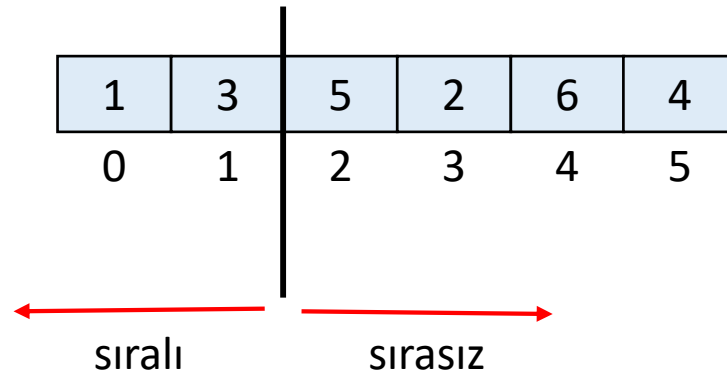


gecici = 1

kaydır



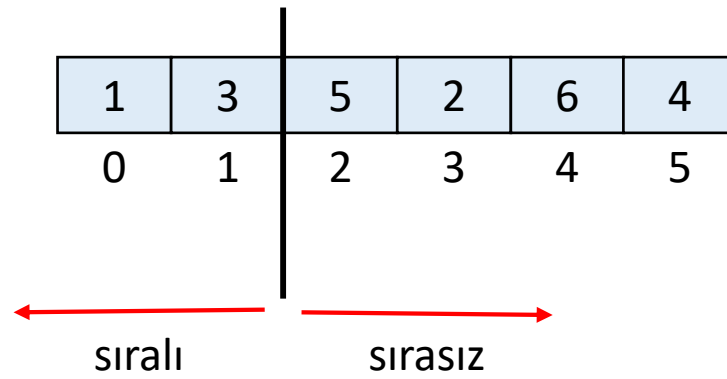
Eklemeli Sıralama



Eklemeli Sıralama



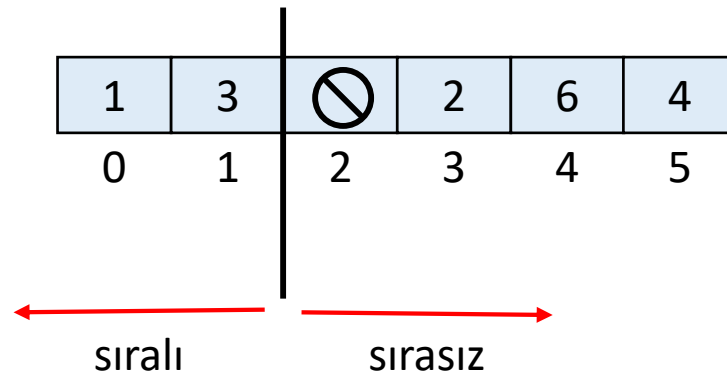
gecici = 5



Eklemeli Sıralama



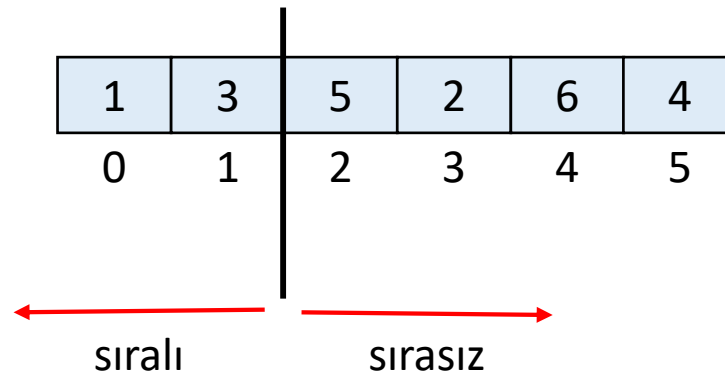
gecici = 5



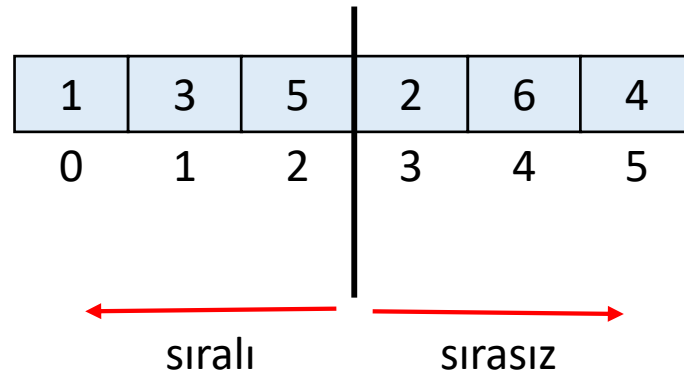
Eklemeli Sıralama



gecici = 5



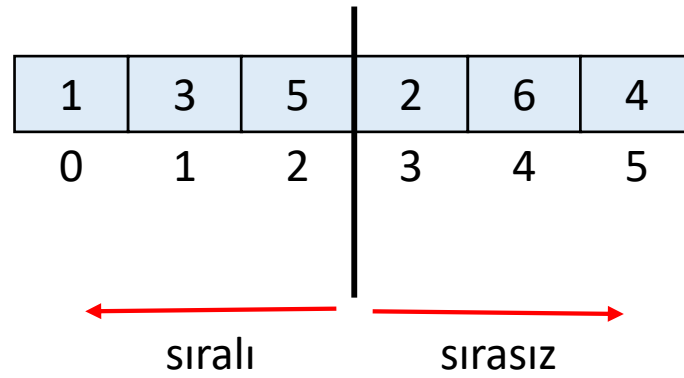
Eklemeli Sıralama



Eklemeli Sıralama



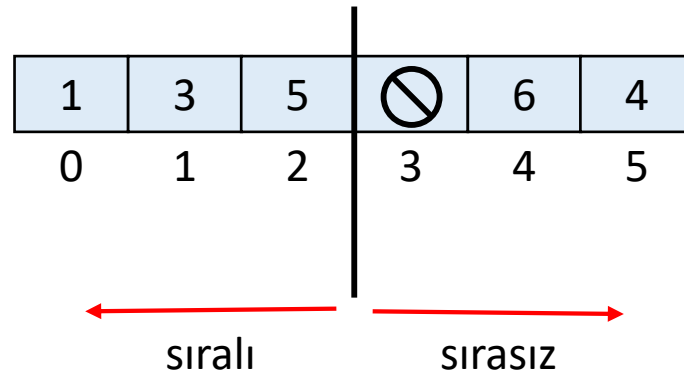
gecici = 2



Eklemeli Sıralama



gecici = 2

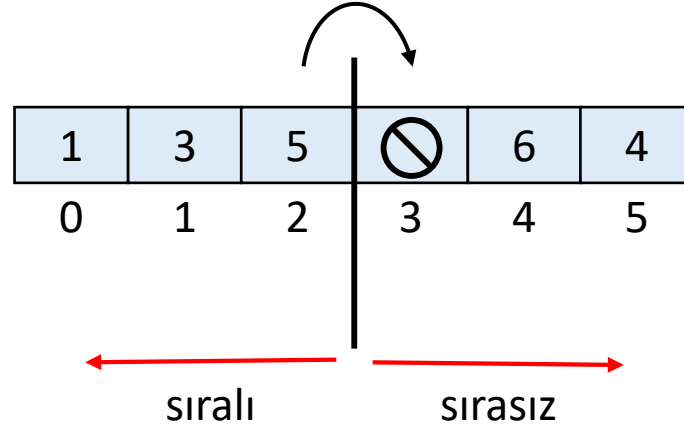


Eklemeli Sıralama



gecici = 2

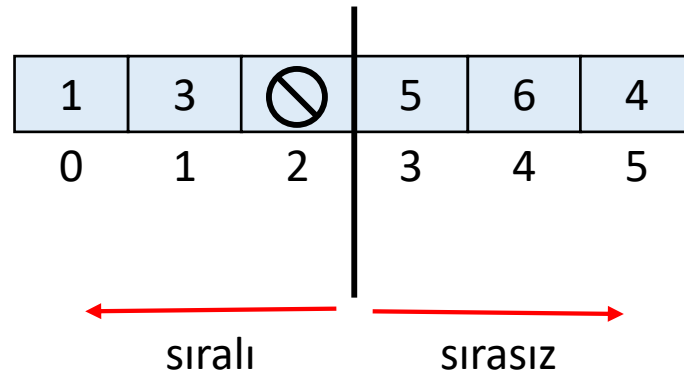
kaydır



Eklemeli Sıralama



gecici = 2

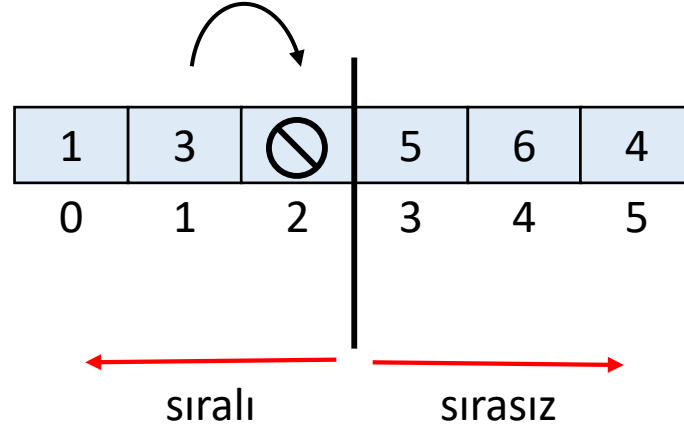


Eklemeli Sıralama



gecici = 2

kaydır

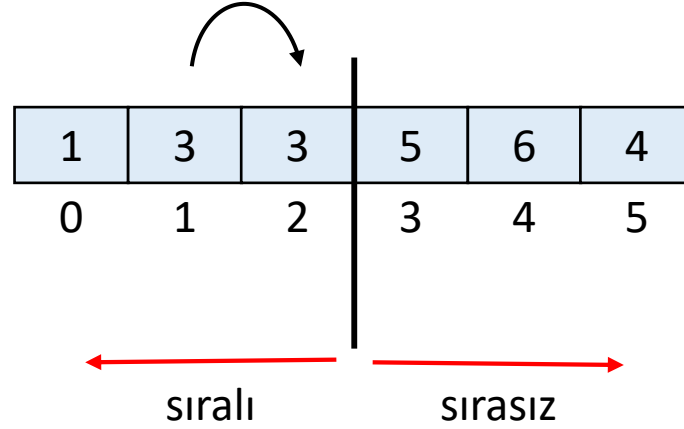


Eklemeli Sıralama



gecici = 2

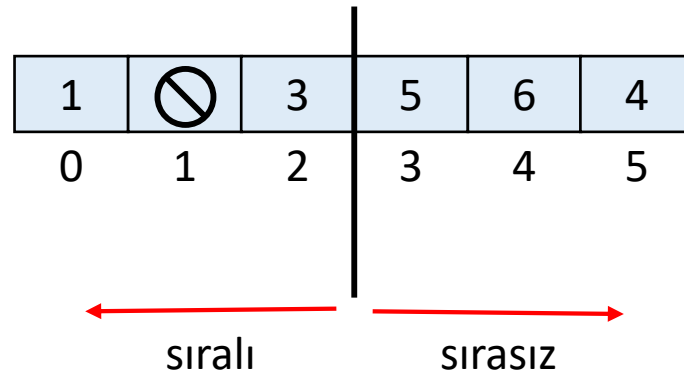
kaydır



Eklemeli Sıralama



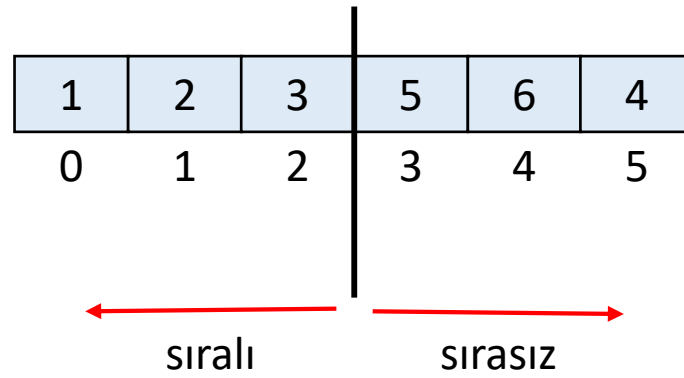
gecici = 2



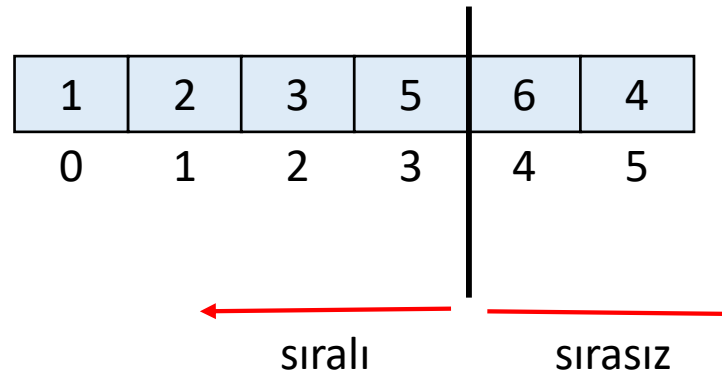
Eklemeli Sıralama



gecici = 2



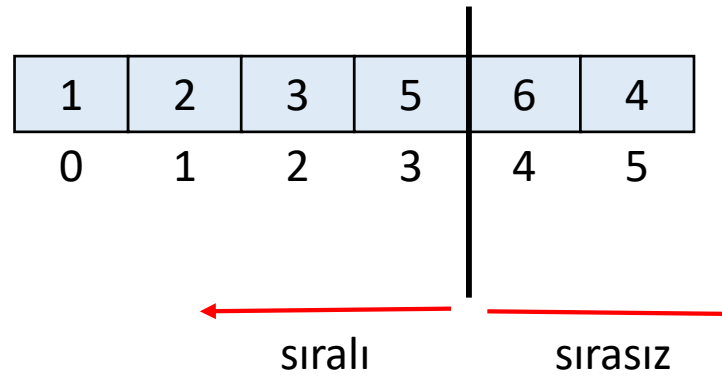
Eklemeli Sıralama



Eklemeli Sıralama



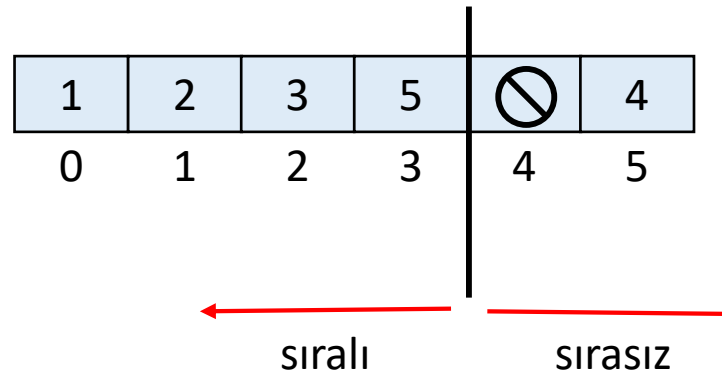
gecici = 6



Eklemeli Sıralama



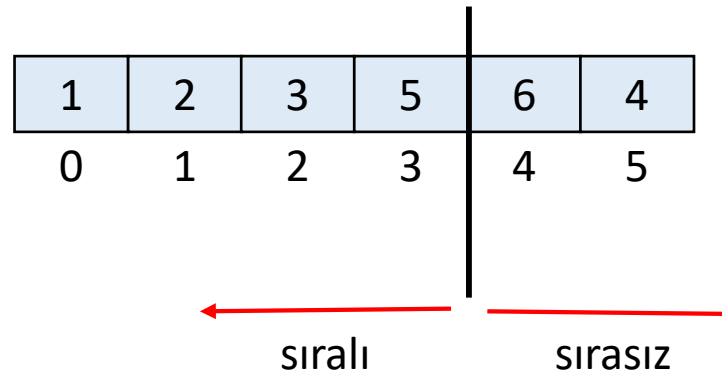
gecici = 6



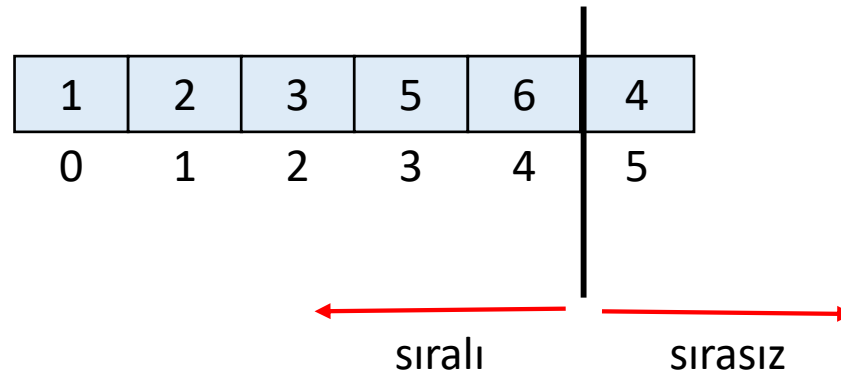
Eklemeli Sıralama



gecici = 6



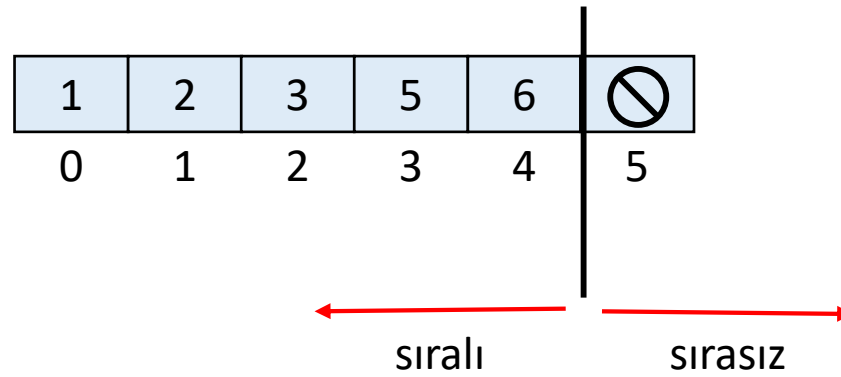
Eklemeli Sıralama



Eklemeli Sıralama



gecici = 4

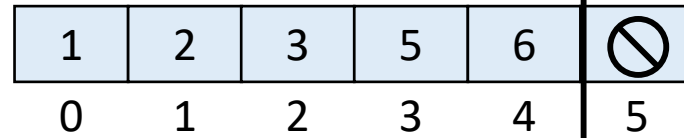


Eklemeli Sıralama



gecici = 4

kaydır



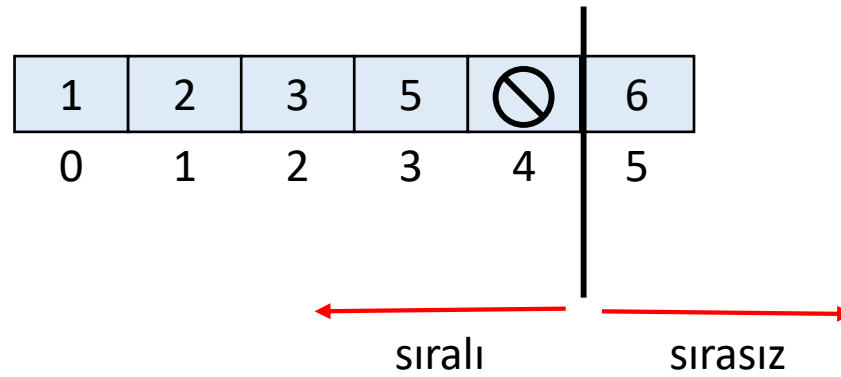
sıralı

sırasız

Eklemeli Sıralama



gecici = 4

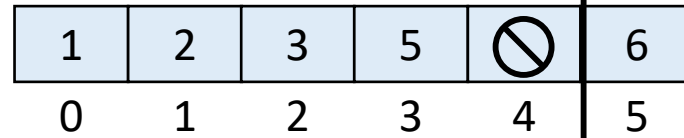


Eklemeli Sıralama



gecici = 4

kaydır

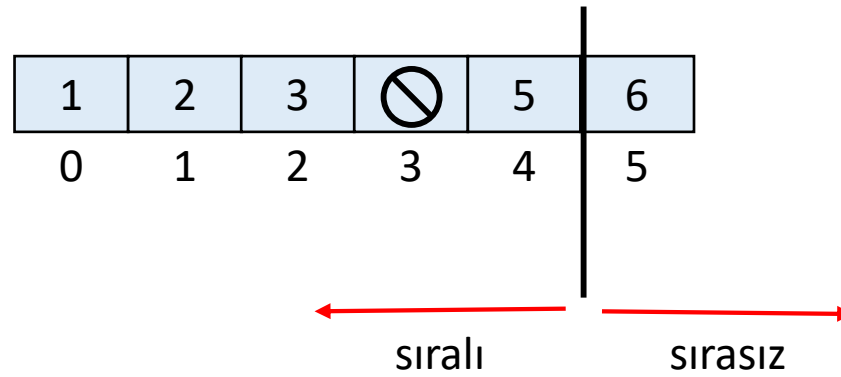


← sıralı → sırasız

Eklemeli Sıralama



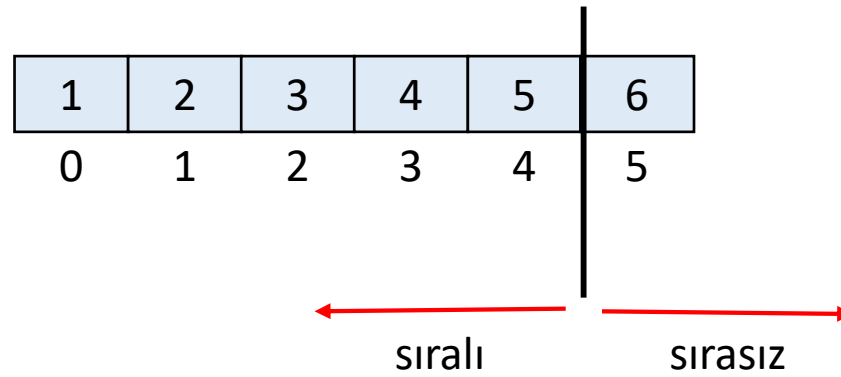
gecici = 4



Eklemeli Sıralama



gecici = 4



Eklemeli Sıralama



1	2	3	4	5	6
0	1	2	3	4	5

Eklemeli Sıralama



dizi[]

5	1	9	2	10
0	1	2	3	4

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



dizi[]

5	1	9	2	10
0	1	2	3	4

→

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



dizi[]

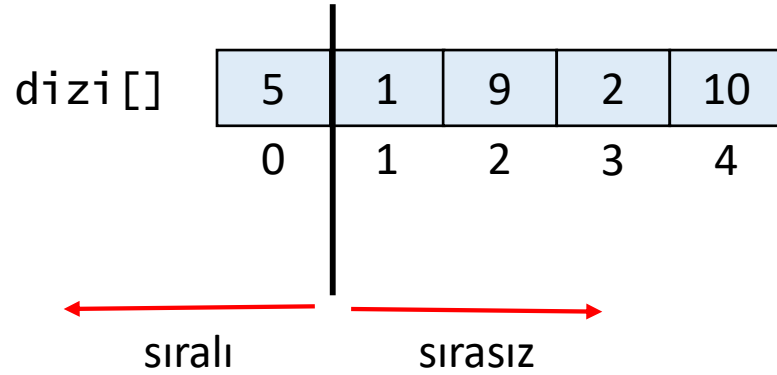
5	1	9	2	10
0	1	2	3	4

n = 5

→

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

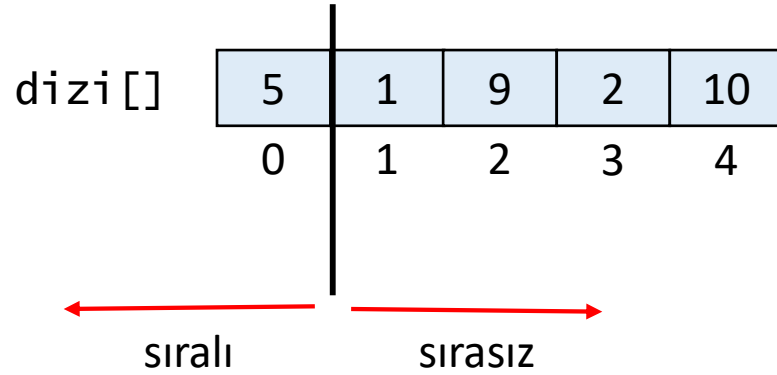

Eklemeli Sıralama



n = 5

```
→ for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama

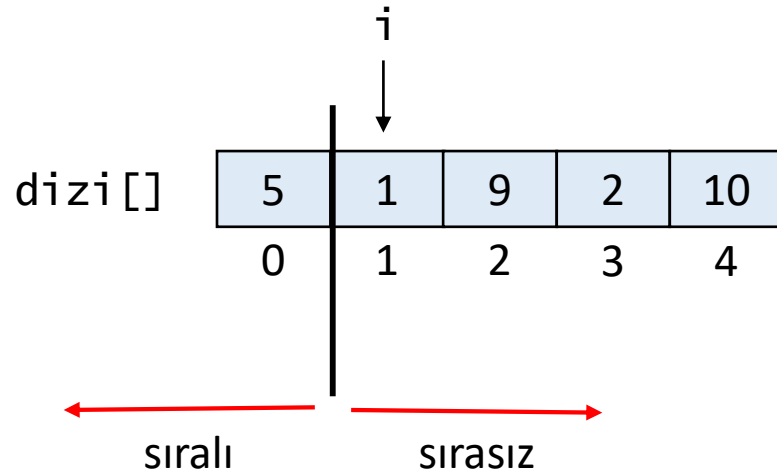


i = 1

n = 5

```
→ for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama

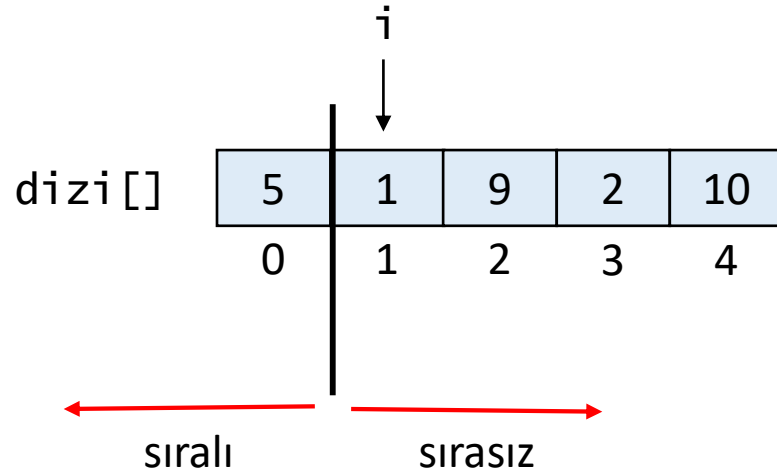


`i = 1`

`n = 5`

```
→ for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



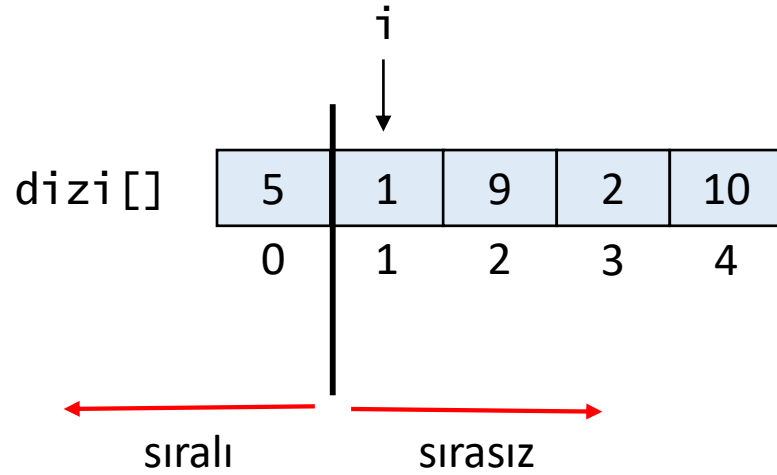
`i = 1`
`gecici = 1`

`n = 5`



```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



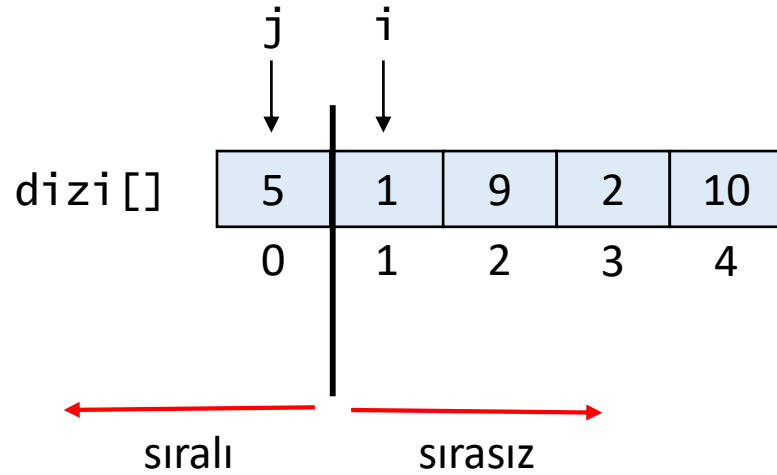
`i = 1`
`gecici = 1`
`j = 0`

`n = 5`



```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



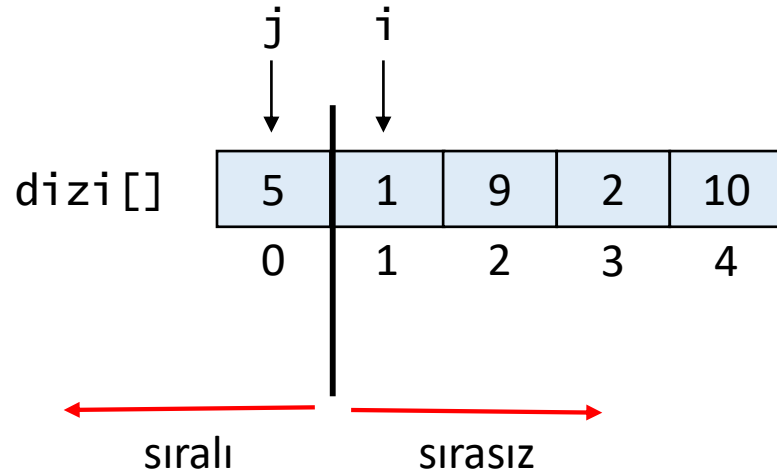
`i = 1`
`gecici = 1`
`j = 0`

`n = 5`



```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



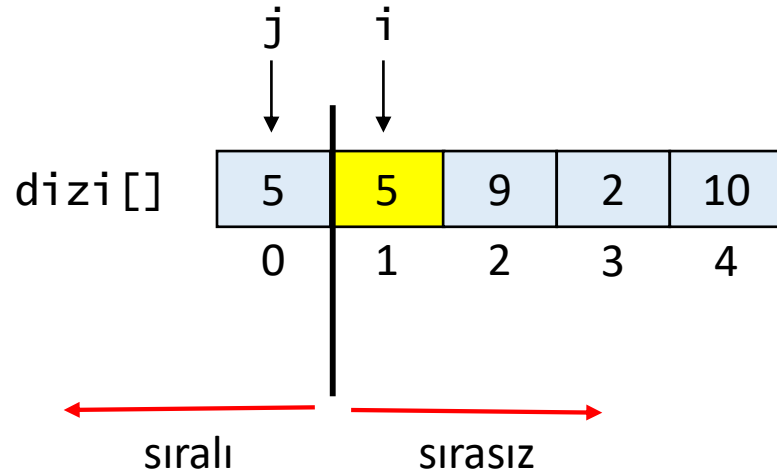
```
i = 1  
gecici = 1  
j = 0
```

```
n = 5
```



```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama

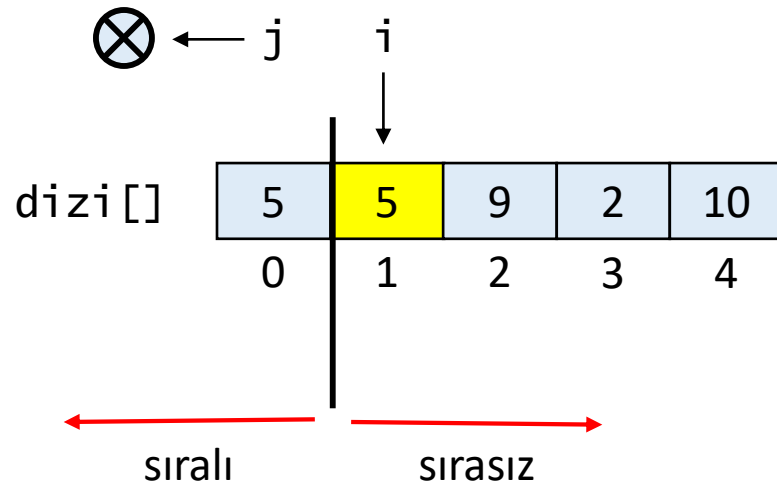


```
i = 1  
gecici = 1  
j = 0
```

```
n = 5
```

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```


Eklemeli Sıralama

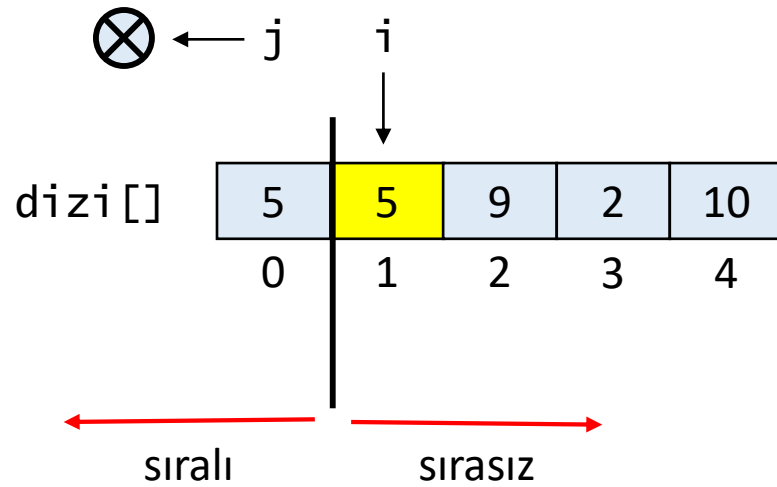


```
i = 1  
gecici = 1  
j = -1
```

```
n = 5
```

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama

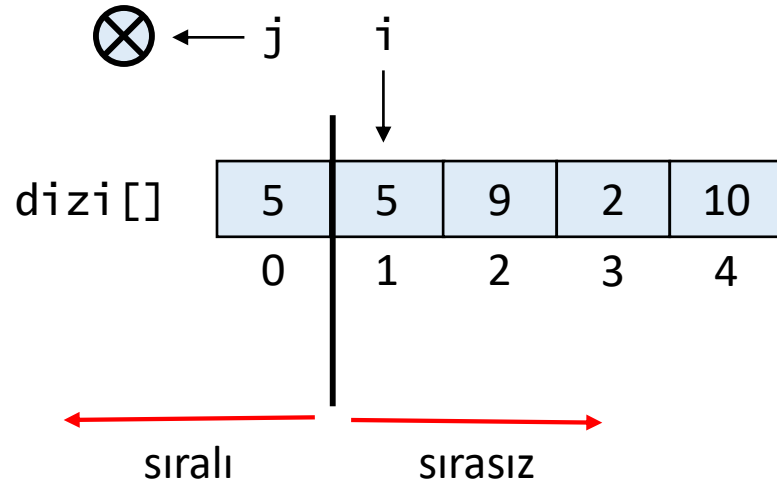


```
i = 1  
gecici = 1  
j = -1
```

```
n = 5
```

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



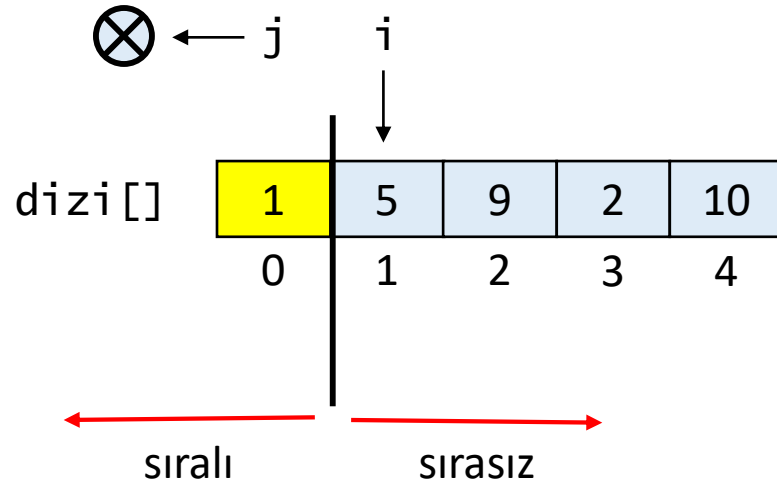
```
i = 1  
gecici = 1  
j = -1
```

```
n = 5
```



```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama

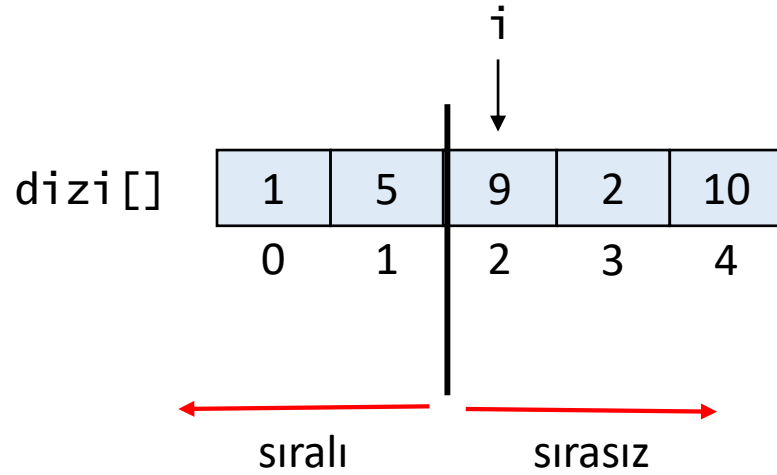


```
i = 1  
gecici = 1  
j = -1
```

```
n = 5
```

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama

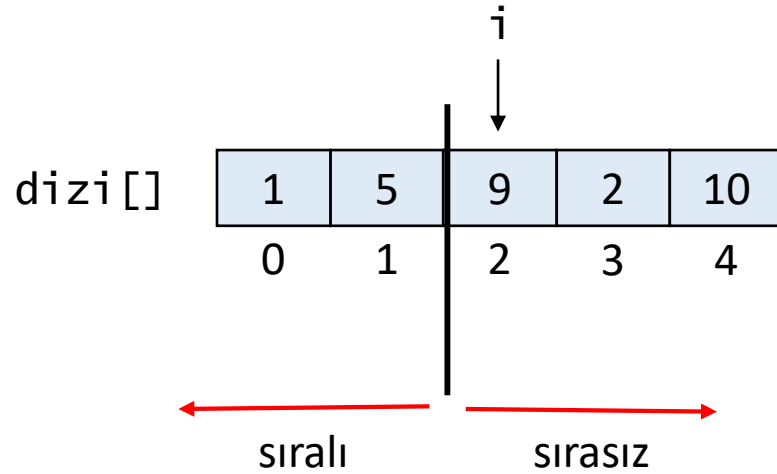


`i = 2`

`n = 5`

```
→ for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



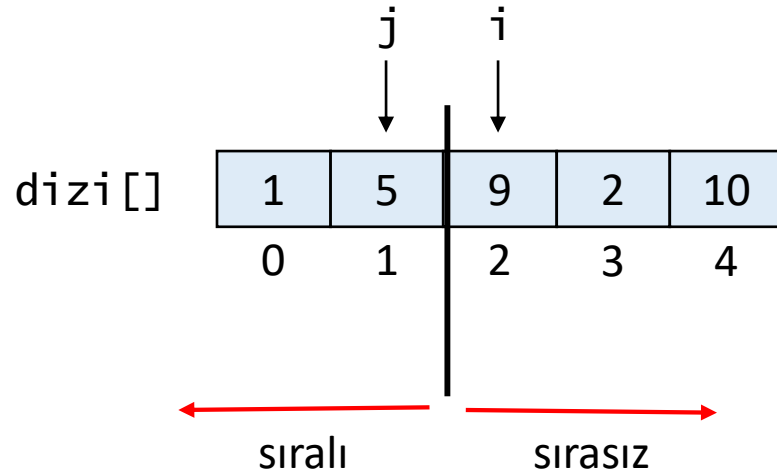
`i = 2`
`gecici = 9`

`n = 5`



```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



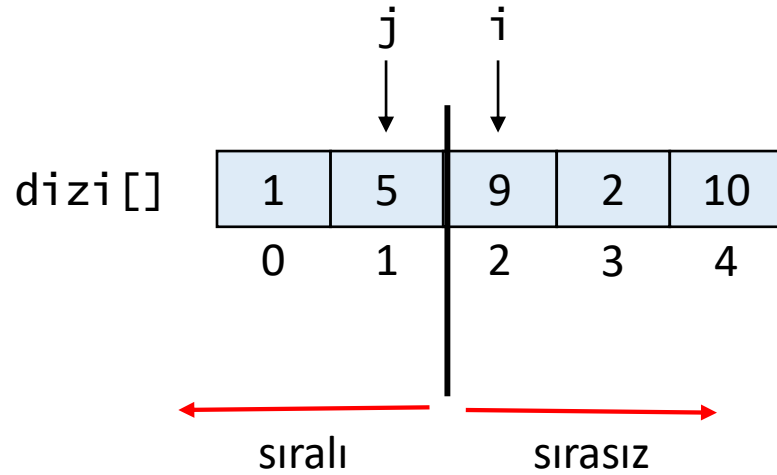
`i = 2`
`gecici = 9`
`j = 1`

`n = 5`



```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama

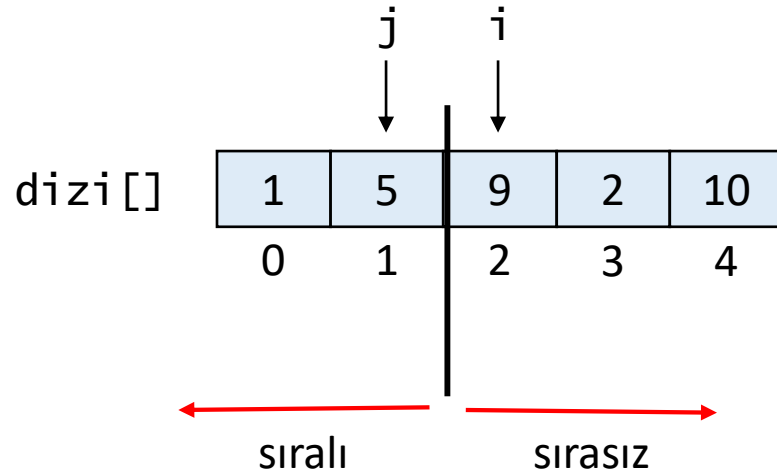


`i = 2`
`gecici = 9`
`j = 1`

`n = 5`

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```


Eklemeli Sıralama

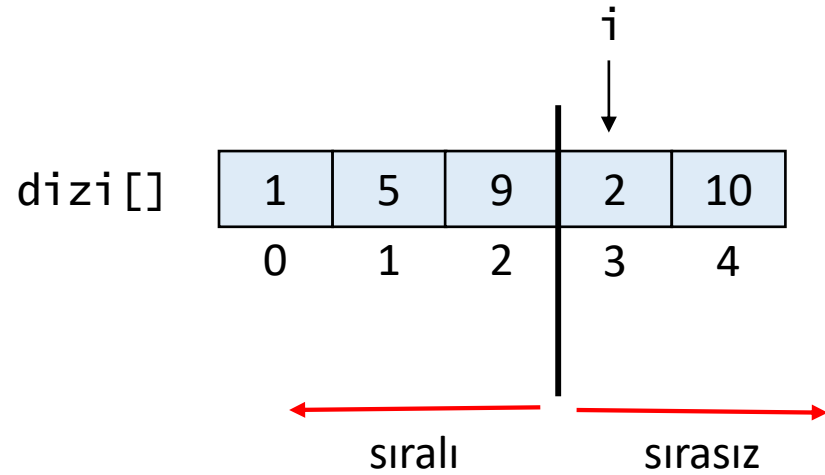


`i = 2`
`gecici = 9`
`j = 1`

`n = 5`

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



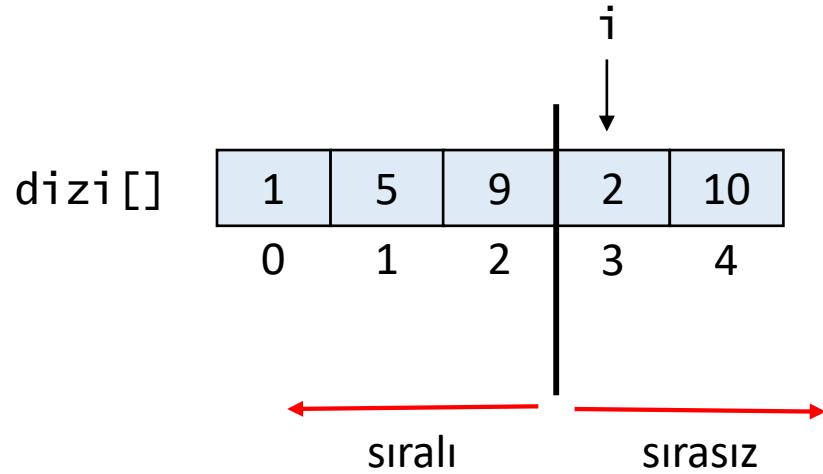
i = 3

n = 5

→

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



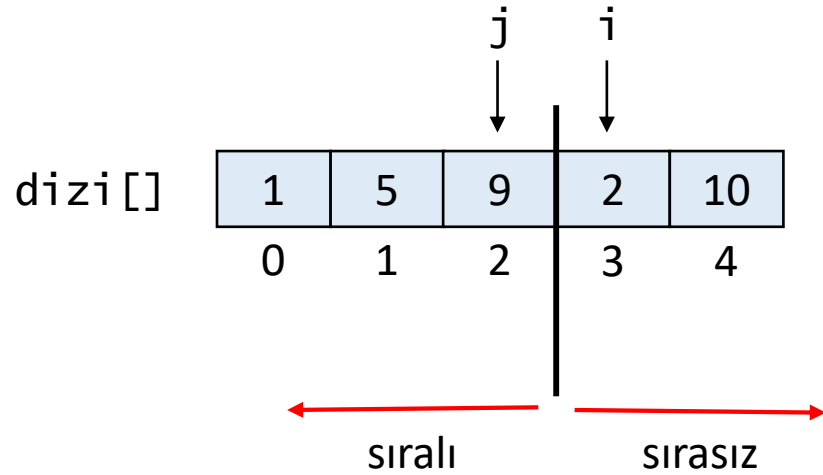
i = 3
gecici = 2

n = 5



```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



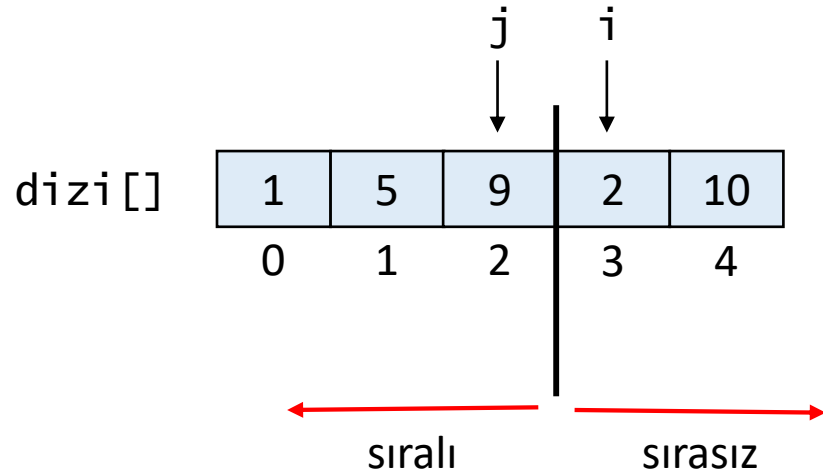
i = 3
gecici = 2
j = 2

n = 5

→

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



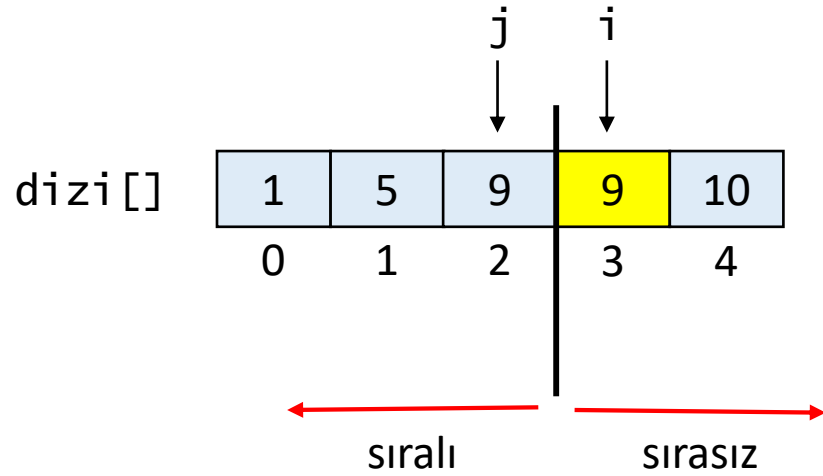
i = 3
gecici = 2
j = 2

n = 5

→

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



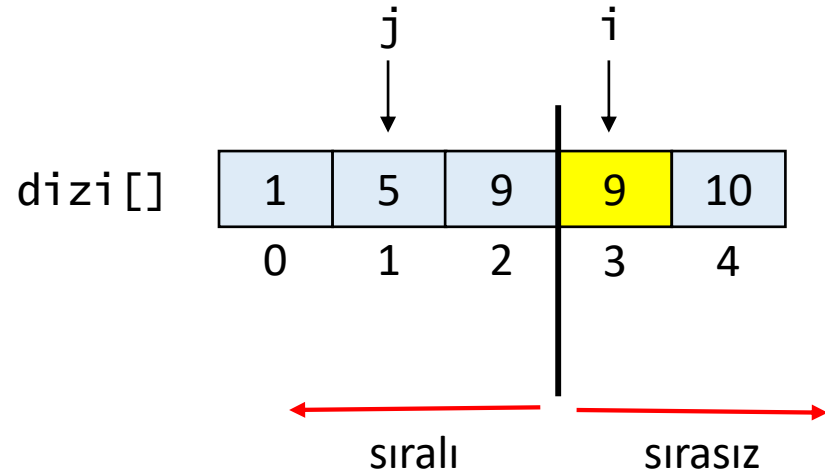
i = 3
gecici = 2
j = 2

n = 5

→

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



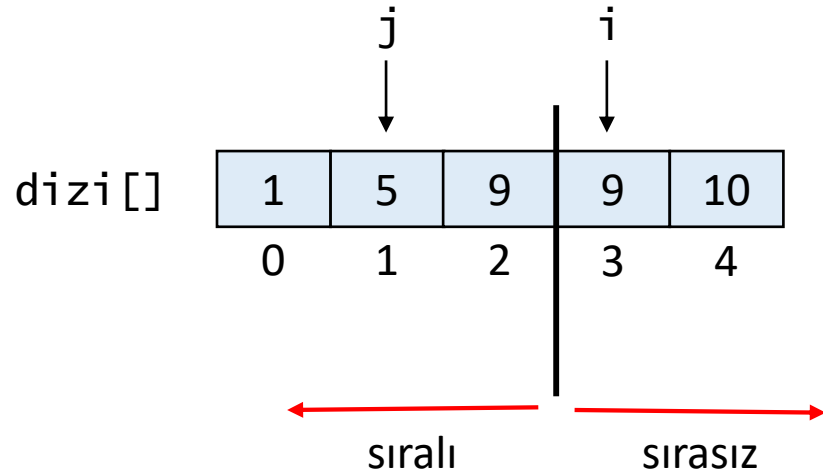
i = 3
gecici = 2
j = 1

n = 5

→

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



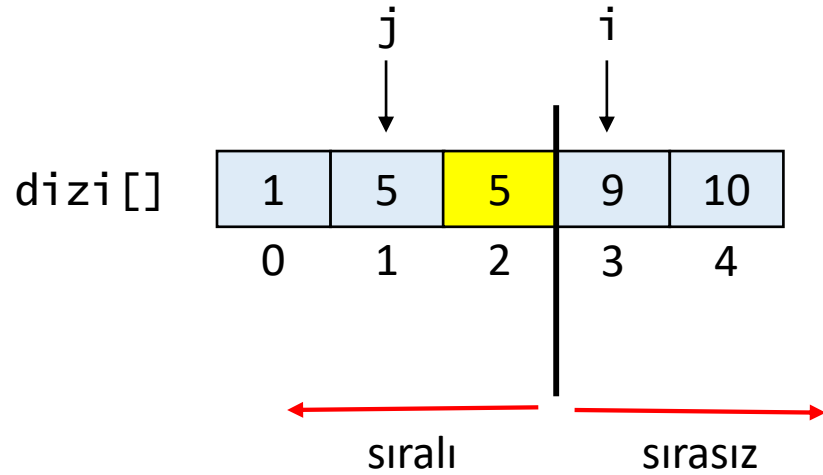
i = 3
gecici = 2
j = 1

n = 5

→

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```


Eklemeli Sıralama



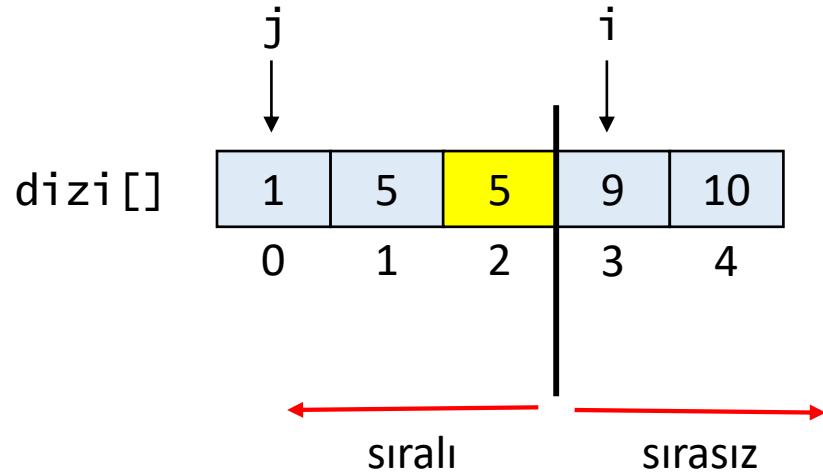
i = 3
gecici = 2
j = 1

n = 5

→

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama

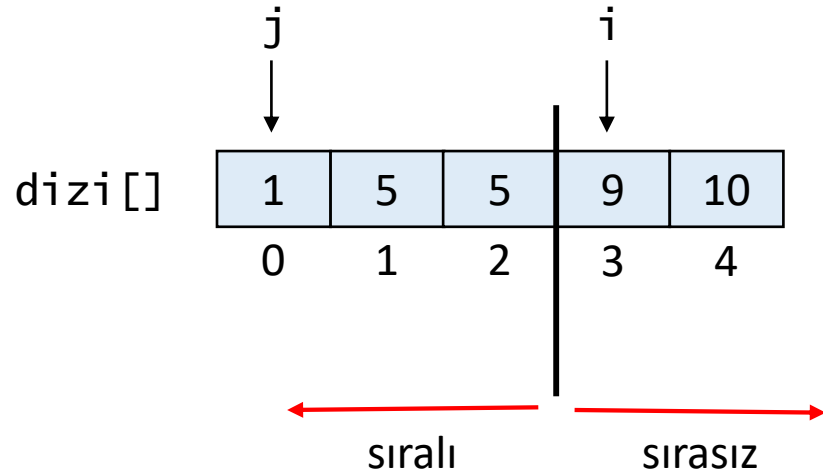


`i = 3`
`gecici = 2`
`j = 0`

`n = 5`

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



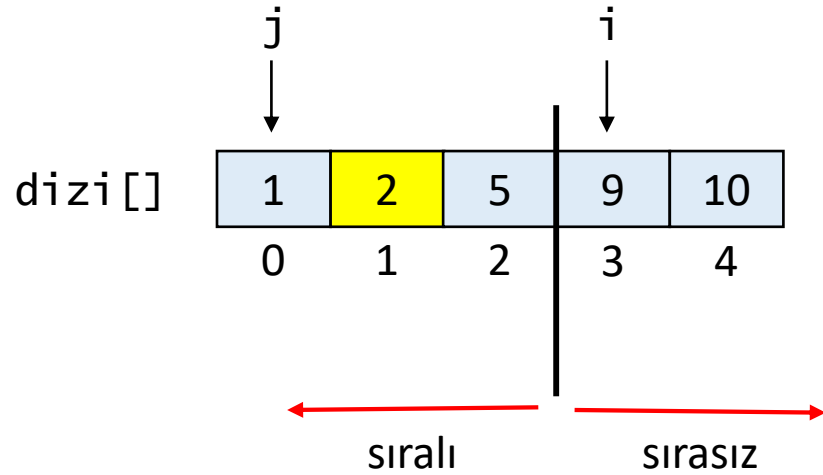
`i = 3`
`gecici = 2`
`j = 0`

`n = 5`

→

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama

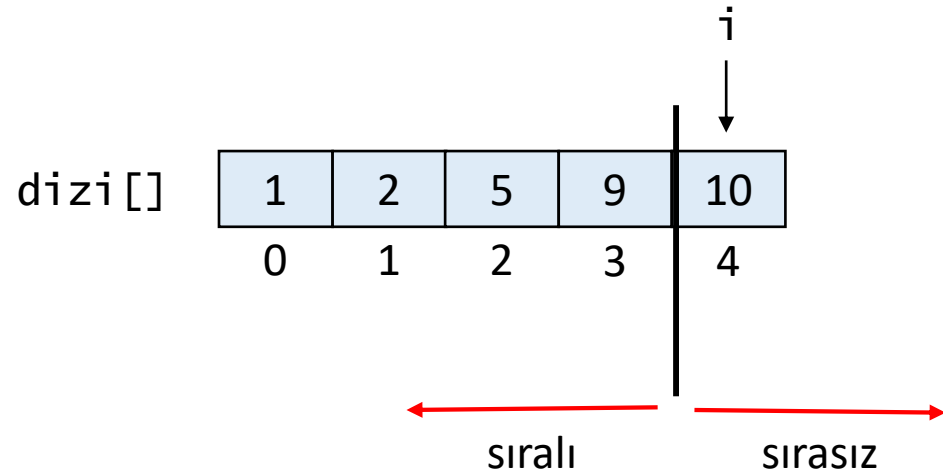


`i = 3`
`gecici = 2`
`j = 0`

`n = 5`

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



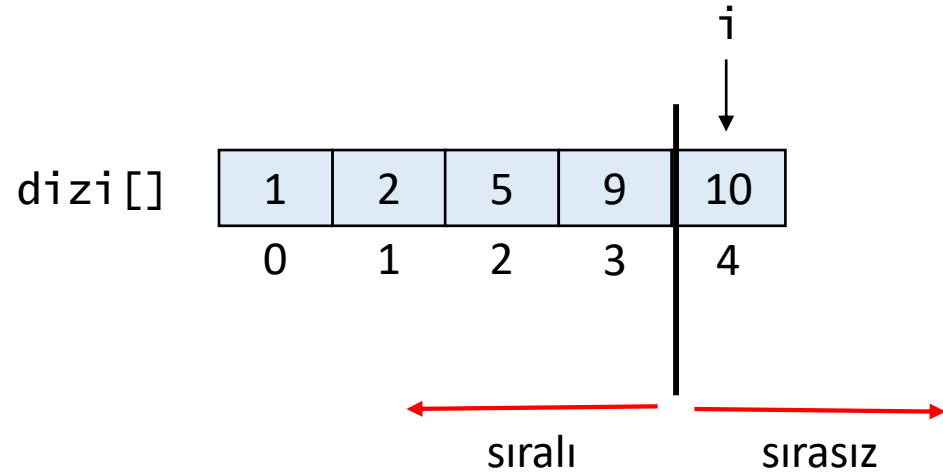
i = 4

n = 5

→

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



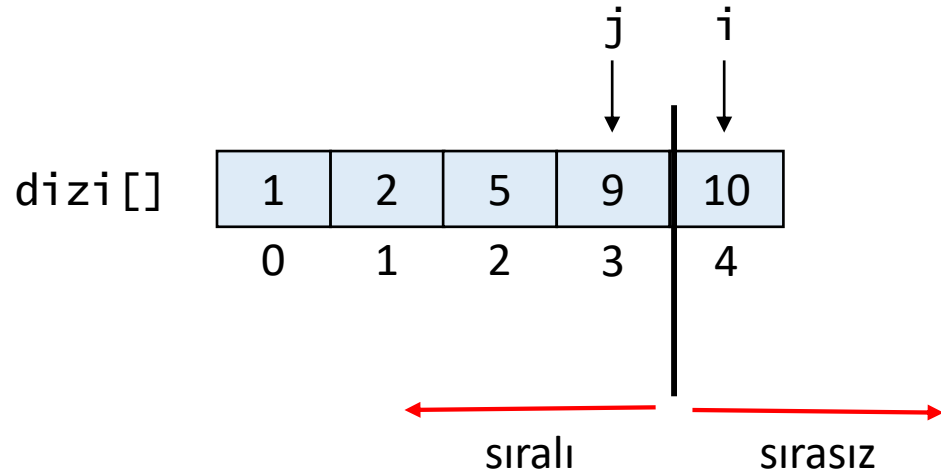
i = 4
gecici = 10

n = 5



```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama

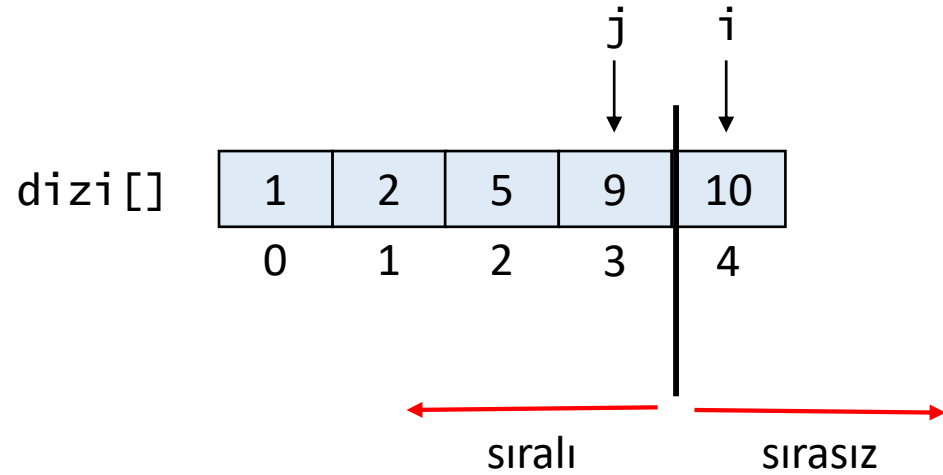


i = 4
gecici = 10
j = 3

n = 5

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



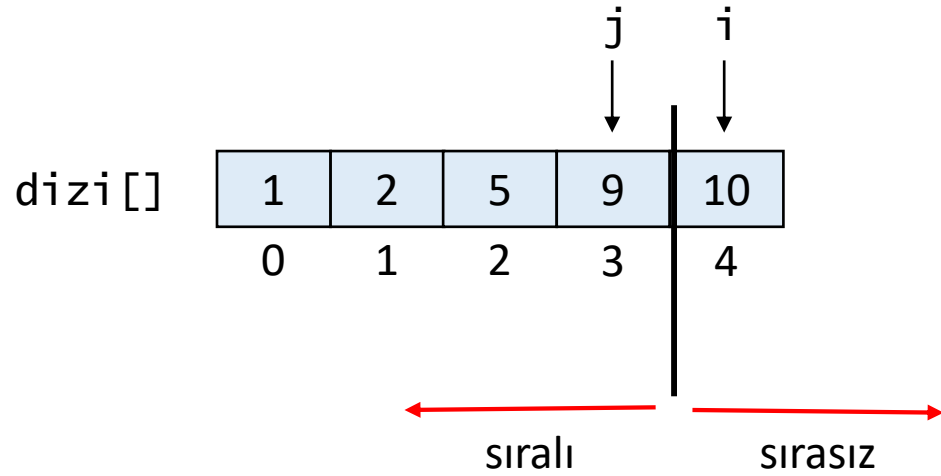
i = 4
gecici = 10
j = 3

n = 5



```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```


Eklemeli Sıralama



`i = 4`
`gecici = 10`
`j = 3`

`n = 5`

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



dizi[]

1	2	5	9	10
0	1	2	3	4

i = 5

n = 5

→

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```

Eklemeli Sıralama



dizi[]

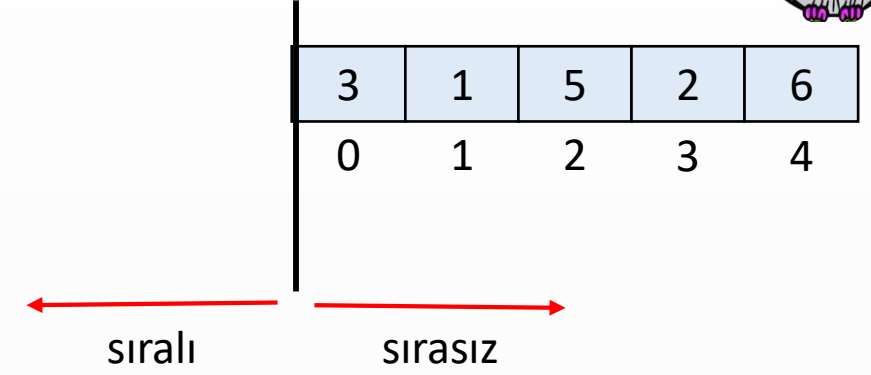
1	2	5	9	10
0	1	2	3	4

```
for(int i = 1; i < n; i++) {  
    int gecici = dizi[i];  
    int j = i - 1;  
    while(j >= 0 && dizi[j] > gecici) {  
        dizi[j+1] = dizi[j];  
        j = j - 1;  
    }  
    dizi[j+1] = gecici;  
}
```



Seçmeli Sıralama (Selection Sort)

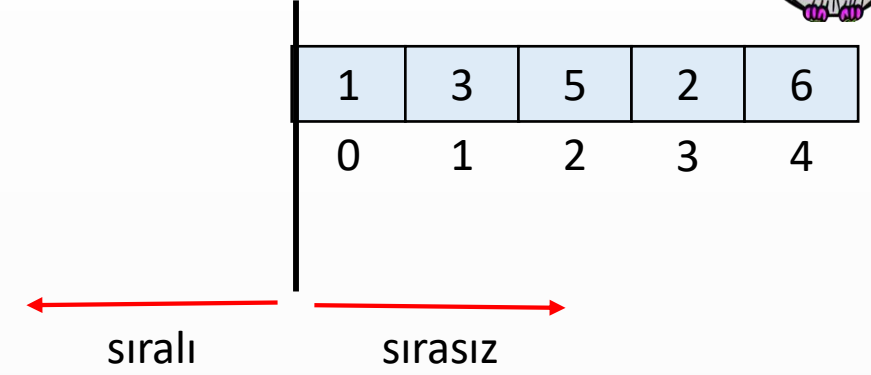
- Verilen dizi iki bölüme ayrılır:
 - Sıralı
 - Sırasız
- Her bir adımda,
 - Sırasız bölümden en küçük eleman bulunur.
 - Sırasız bölümün başındaki eleman ile yer değiştirilir.
 - Yer değiştirmenin ardından sıralı bölümün parçası olur.





Seçmeli Sıralama (Selection Sort)

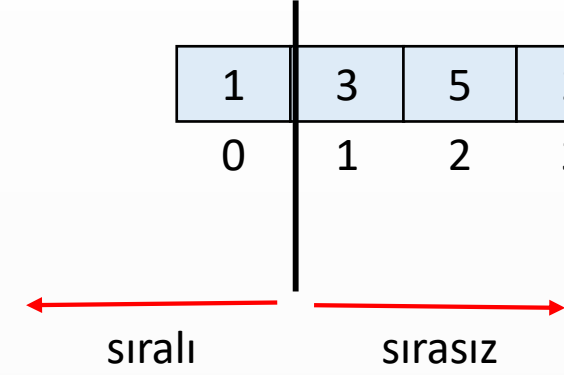
- Verilen dizi iki bölüme ayrılır:
 - Sıralı
 - Sırasız
- Her bir adımda,
 - Sırasız bölümden en küçük eleman bulunur.
 - Sırasız bölümün başındaki eleman ile yer değiştirilir.
 - Yer değiştirmenin ardından sıralı bölümün parçası olur.





Seçmeli Sıralama (Selection Sort)

- Verilen dizi iki bölüme ayrılır:
 - Sıralı
 - Sırasız
- Her bir adımda,
 - Sırasız bölümden en küçük eleman bulunur.
 - Sırasız bölümün başındaki eleman ile yer değiştirilir.
 - Yer değiştirmenin ardından sıralı bölümün parçası olur.



Seçmeli Sıralama



3	1	5	2	6	4
0	1	2	3	4	5

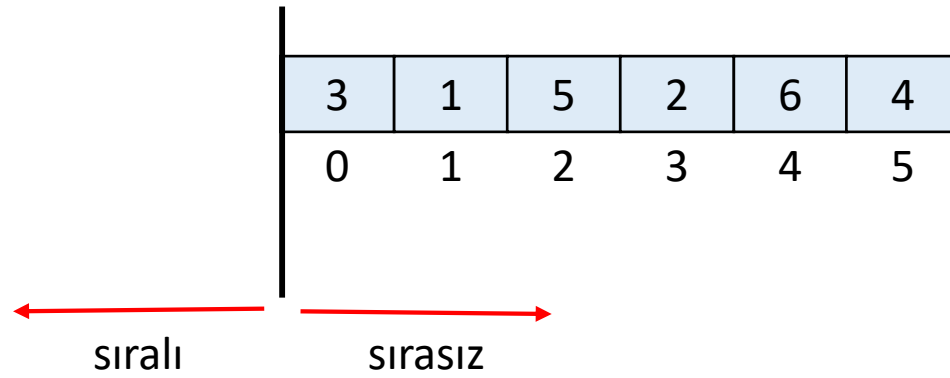
Seçmeli Sıralama



3	1	5	2	6	4
0	1	2	3	4	5

uzunluk = 6

Seçmeli Sıralama

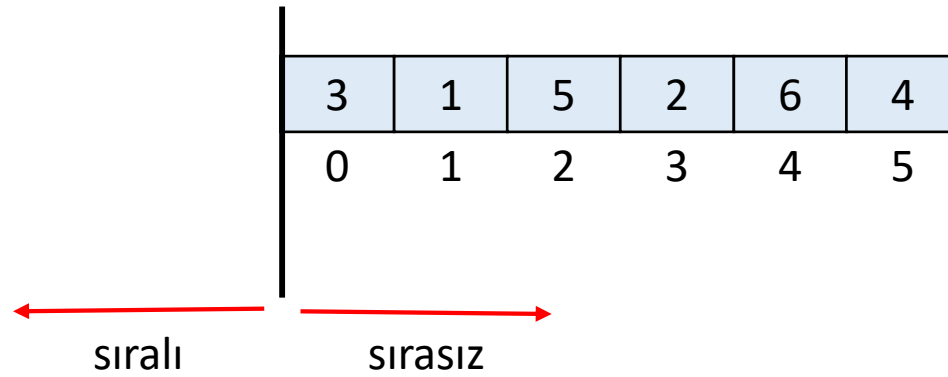


uzunluk = 6

Seçmeli Sıralama



1. Tur



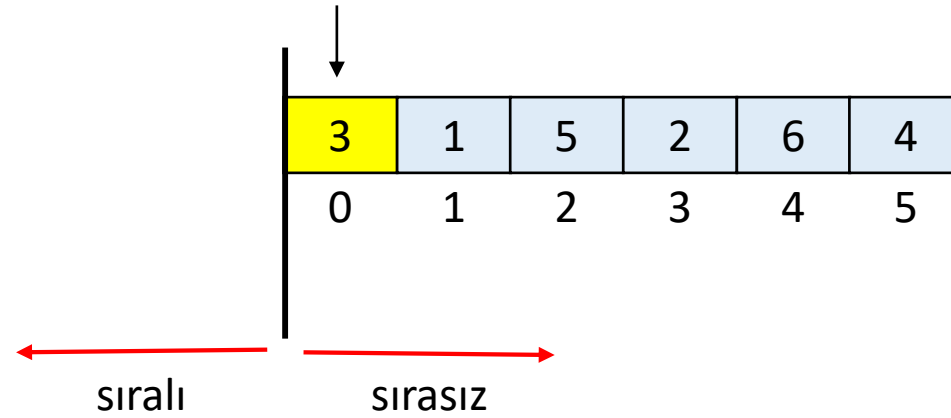
uzunluk = 6

Seçmeli Sıralama



1. Tur

$\text{min} = 3$



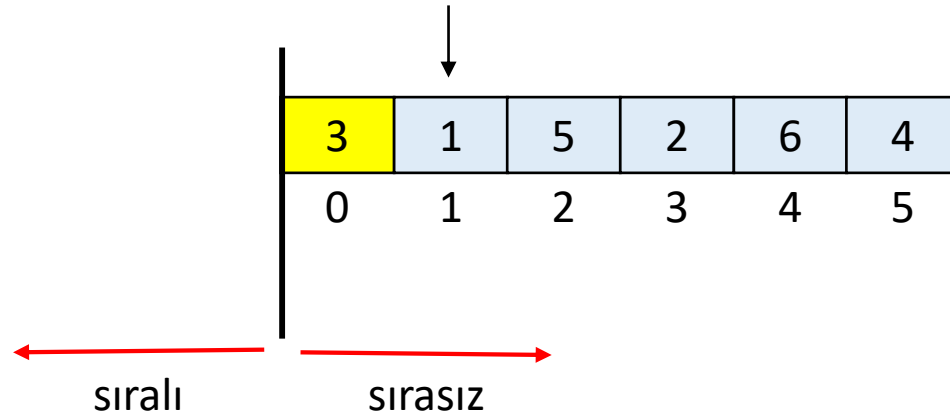
$\text{uzunluk} = 6$

Seçmeli Sıralama



1. Tur

$\text{min} = 3$
 $1 < \text{min} ?$



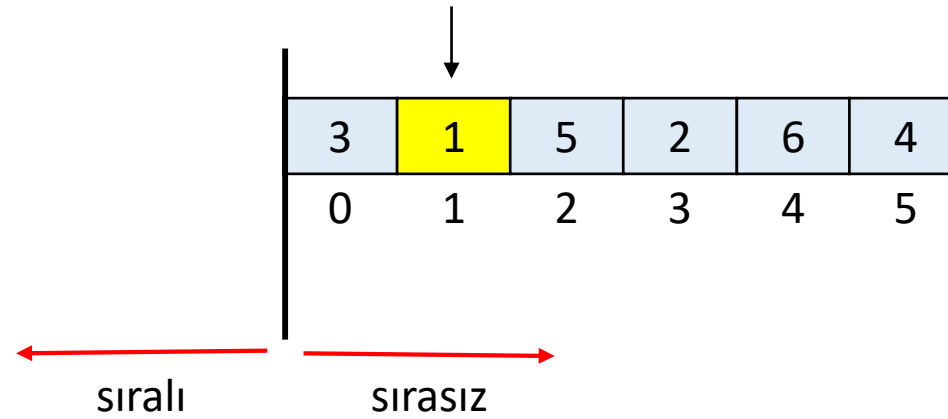
uzunluk = 6

Seçmeli Sıralama



1. Tur

$\text{min} = 1$



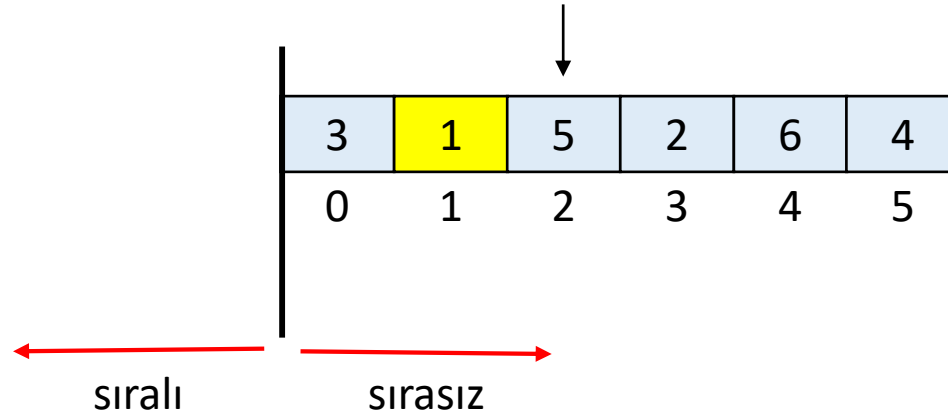
uzunluk = 6

Seçmeli Sıralama



1. Tur

$\text{min} = 1$
 $5 < \text{min} ?$



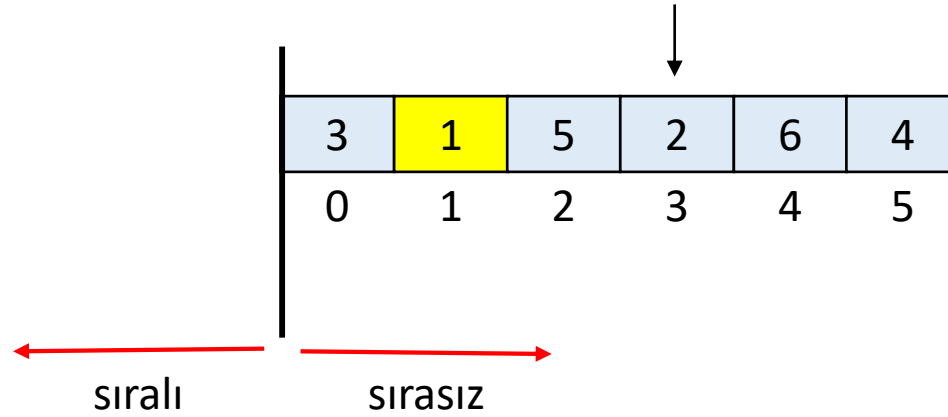
uzunluk = 6

Seçmeli Sıralama



1. Tur

$\text{min} = 1$
 $2 < \text{min} ?$



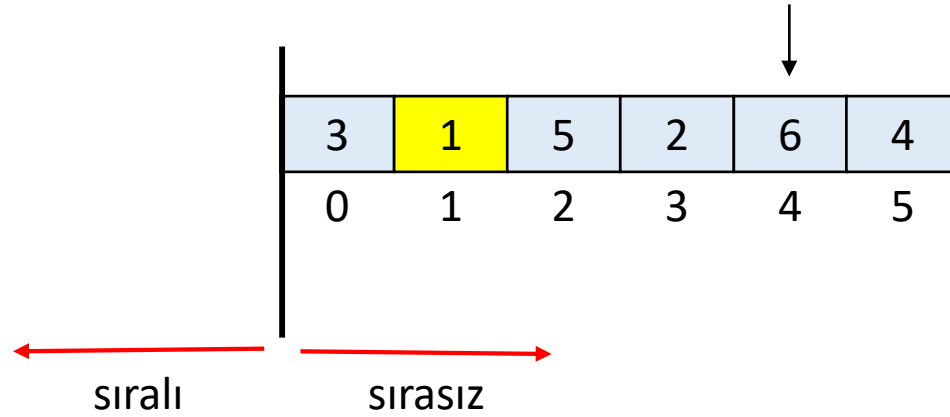
uzunluk = 6

Seçmeli Sıralama



1. Tur

$\text{min} = 1$
 $6 < \text{min} ?$



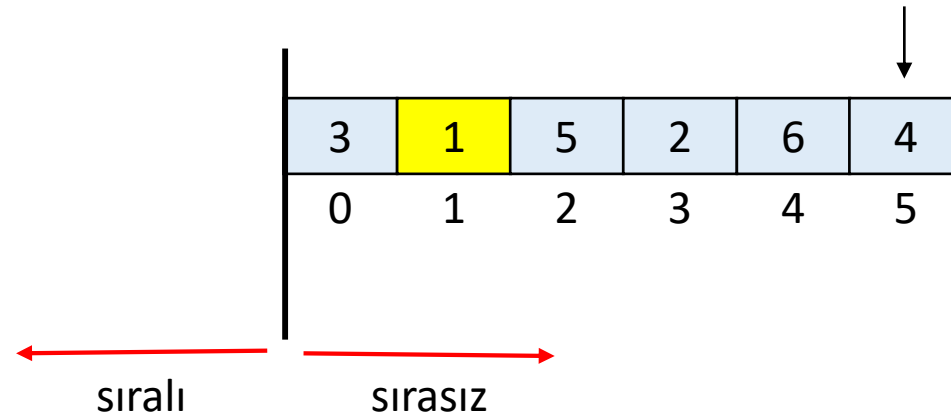
uzunluk = 6

Seçmeli Sıralama



1. Tur

$\text{min} = 1$
 $4 < \text{min} ?$



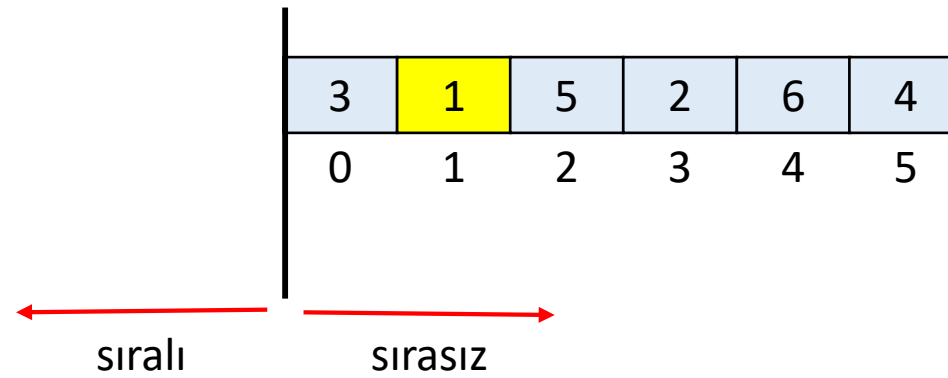
uzunluk = 6

Seçmeli Sıralama



1. Tur

$\text{min} = 1$



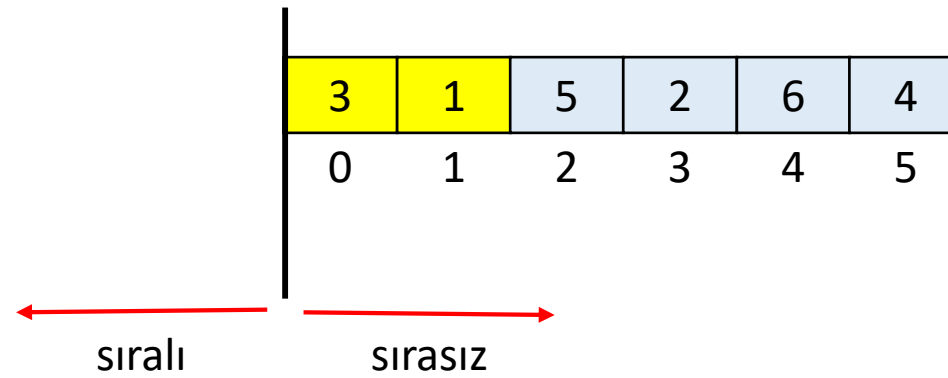
$\text{uzunluk} = 6$

Seçmeli Sıralama



1. Tur

$\text{min} = 1$



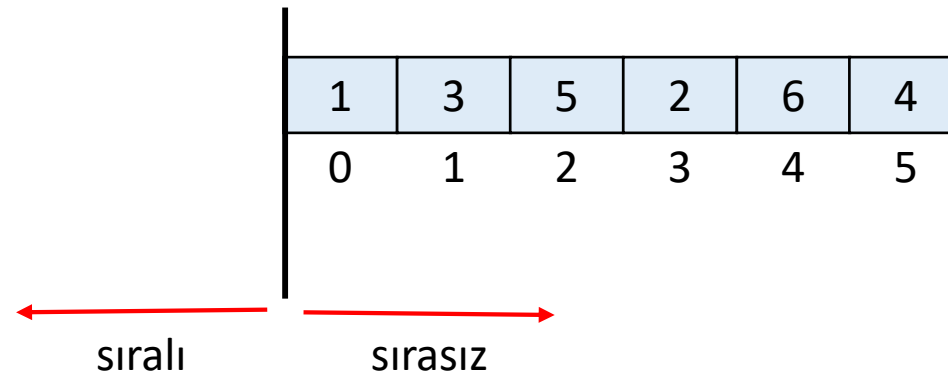
uzunluk = 6

Seçmeli Sıralama



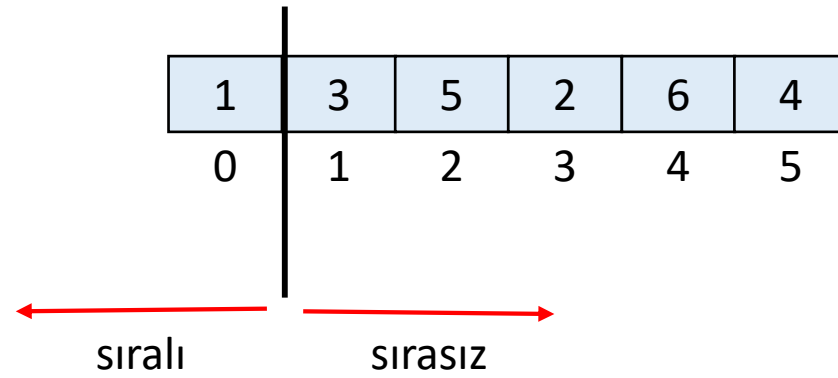
1. Tur

$\text{min} = 1$



uzunluk = 6

Seçmeli Sıralama

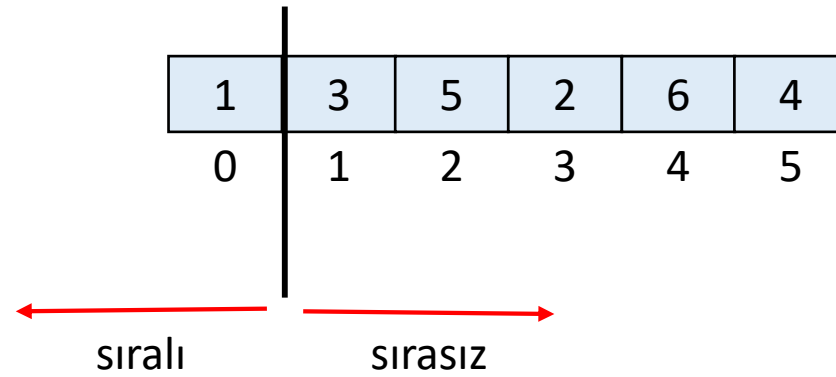


uzunluk = 6

Seçmeli Sıralama



2. Tur



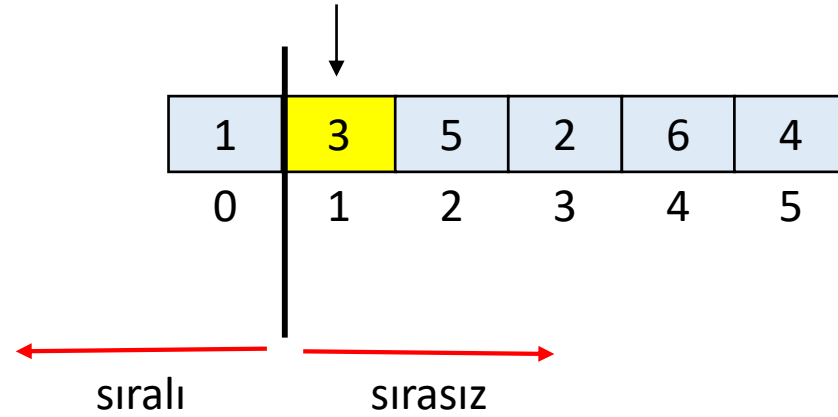
uzunluk = 6

Seçmeli Sıralama



2. Tur

min = 3



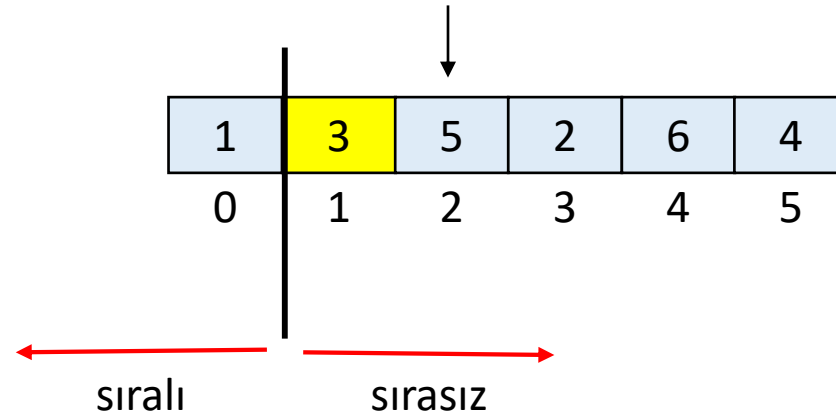
uzunluk = 6

Seçmeli Sıralama



2. Tur

$\text{min} = 3$
 $5 < \text{min} ?$



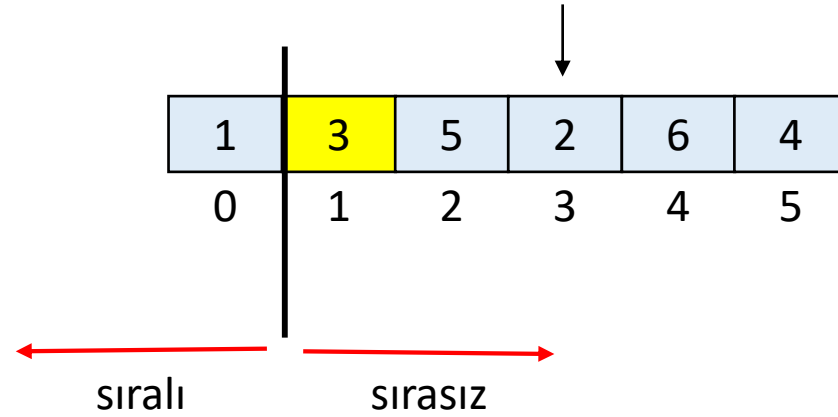
uzunluk = 6

Seçmeli Sıralama



2. Tur

$\text{min} = 3$
 $2 < \text{min} ?$



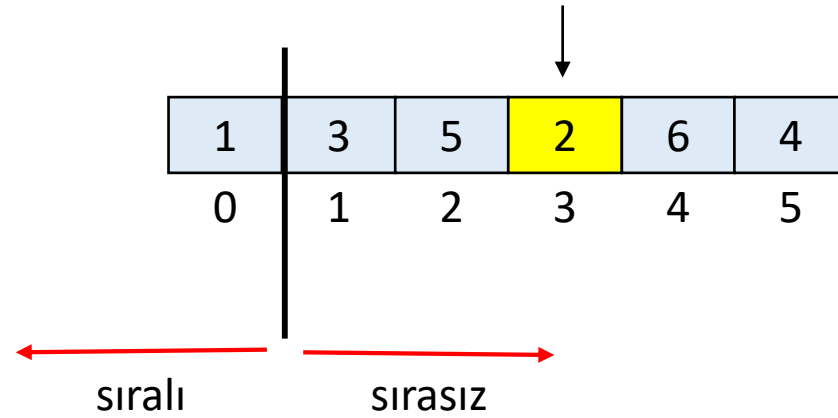
uzunluk = 6

Seçmeli Sıralama



2. Tur

$\text{min} = 2$



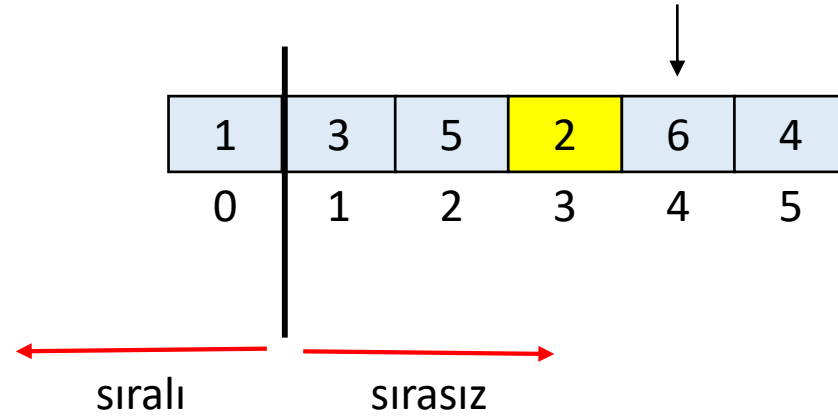
uzunluk = 6

Seçmeli Sıralama



2. Tur

$\text{min} = 2$
 $6 < \text{min} ?$



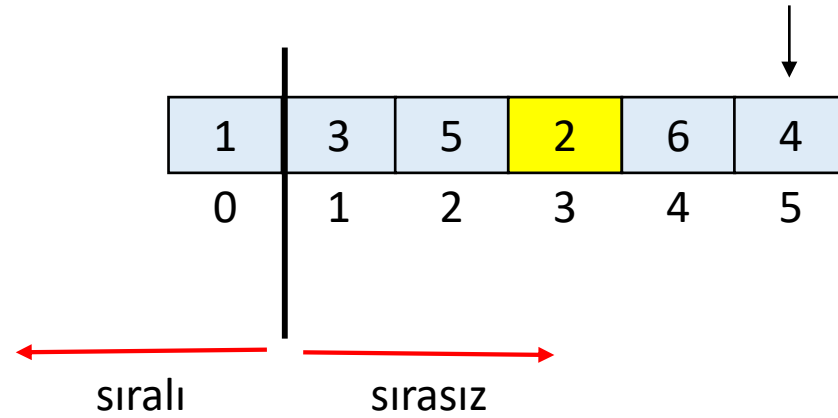
uzunluk = 6

Seçmeli Sıralama



2. Tur

$\text{min} = 2$
 $4 < \text{min} ?$



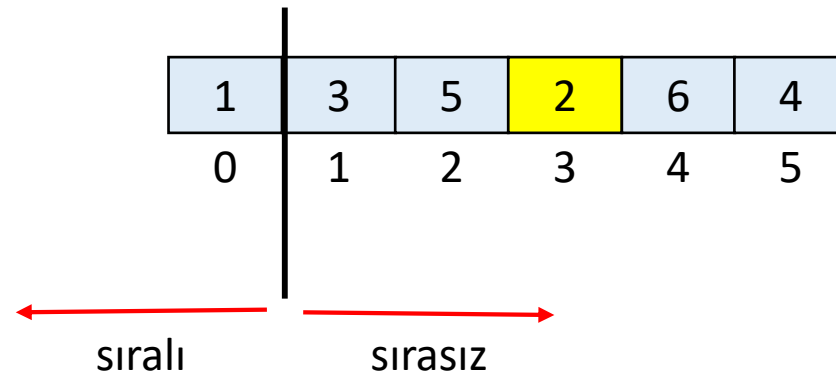
uzunluk = 6

Seçmeli Sıralama



2. Tur

$\text{min} = 2$



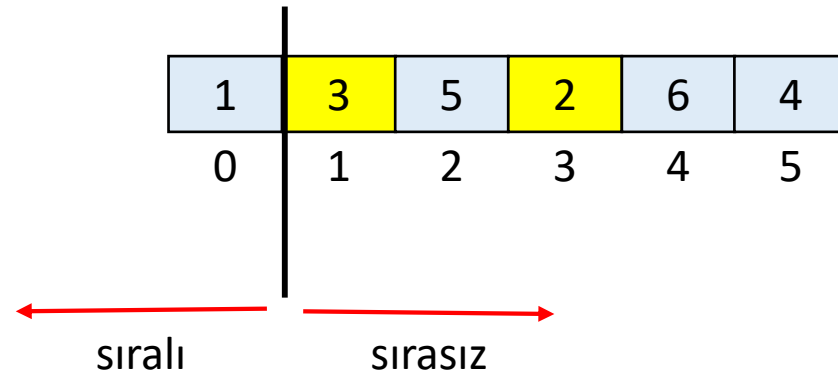
uzunluk = 6

Seçmeli Sıralama



2. Tur

min = 2



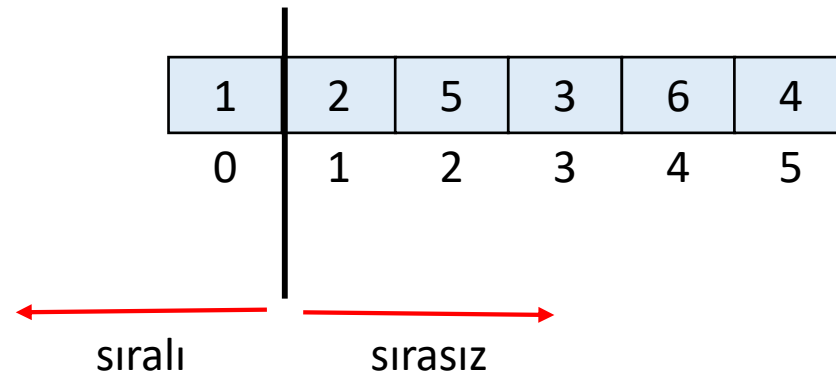
uzunluk = 6

Seçmeli Sıralama



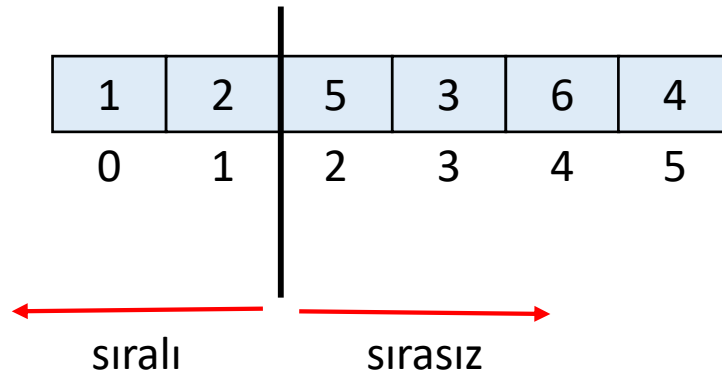
2. Tur

$\text{min} = 2$



uzunluk = 6

Seçmeli Sıralama

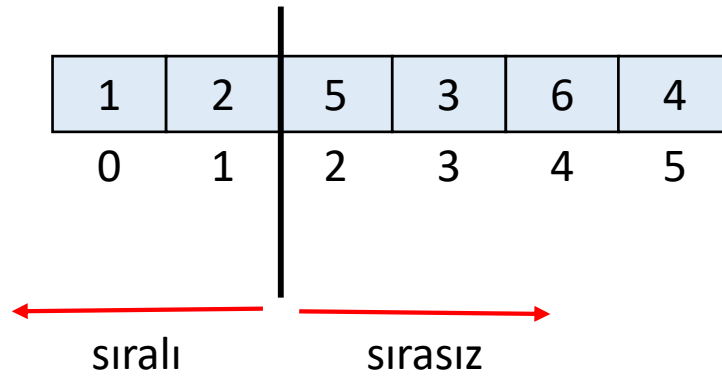


uzunluk = 6

Seçmeli Sıralama



3. Tur



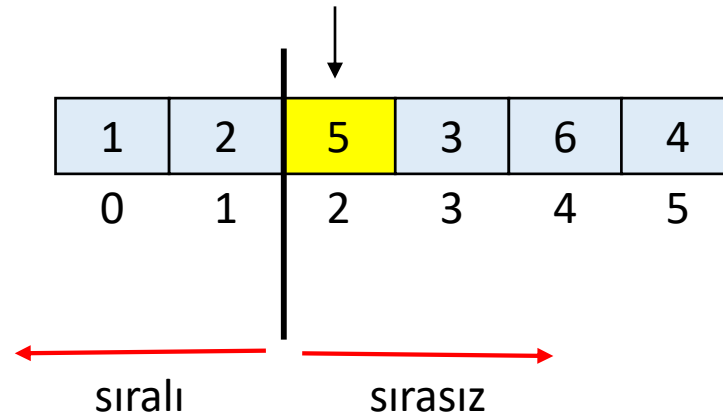
uzunluk = 6

Seçmeli Sıralama



3. Tur

min = 5



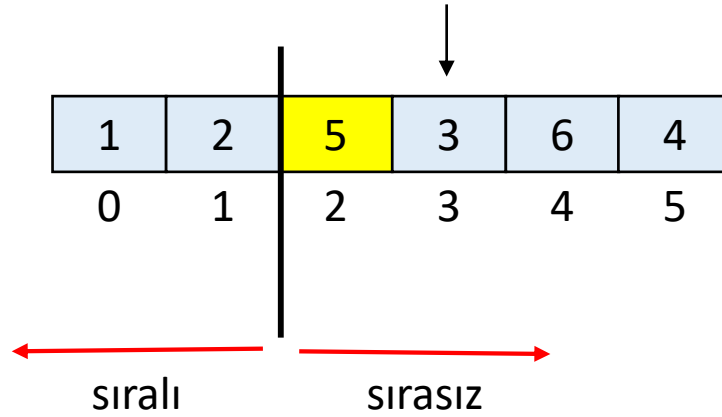
uzunluk = 6

Seçmeli Sıralama



3. Tur

$\text{min} = 5$
 $3 < \text{min} ?$



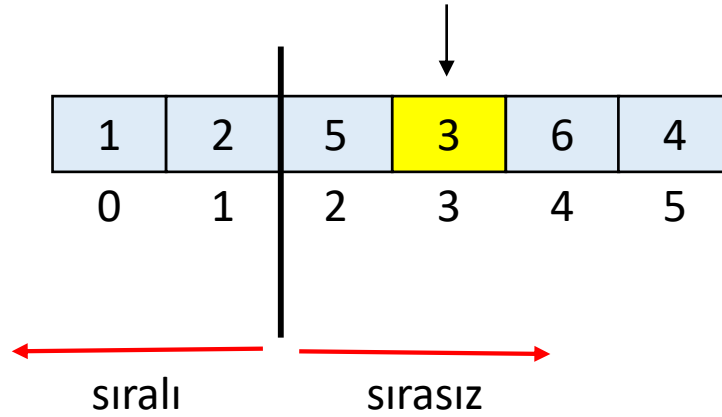
uzunluk = 6

Seçmeli Sıralama



3. Tur

$\text{min} = 3$



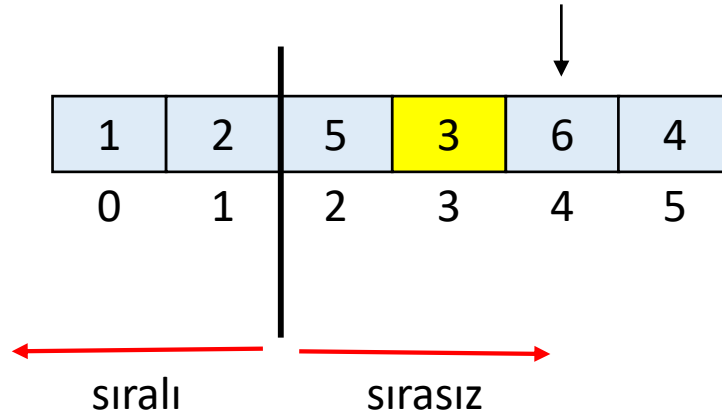
uzunluk = 6

Seçmeli Sıralama



3. Tur

$\text{min} = 3$
 $6 < \text{min} ?$



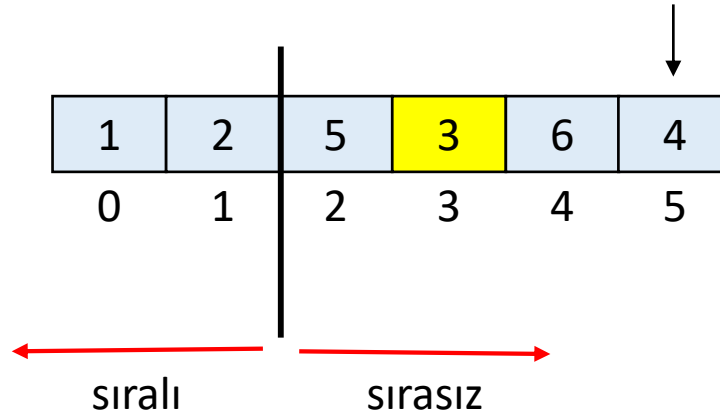
uzunluk = 6

Seçmeli Sıralama



3. Tur

$\text{min} = 3$
 $4 < \text{min} ?$



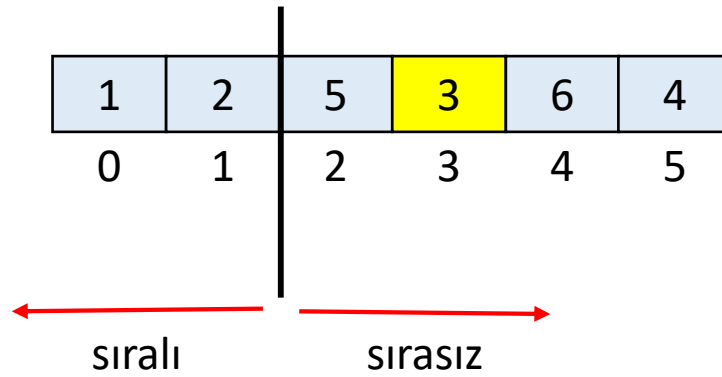
uzunluk = 6

Seçmeli Sıralama



3. Tur

$\text{min} = 3$



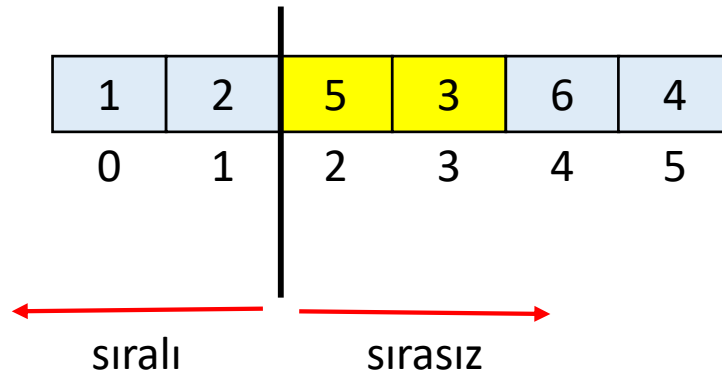
$\text{uzunluk} = 6$

Seçmeli Sıralama



3. Tur

$\text{min} = 3$



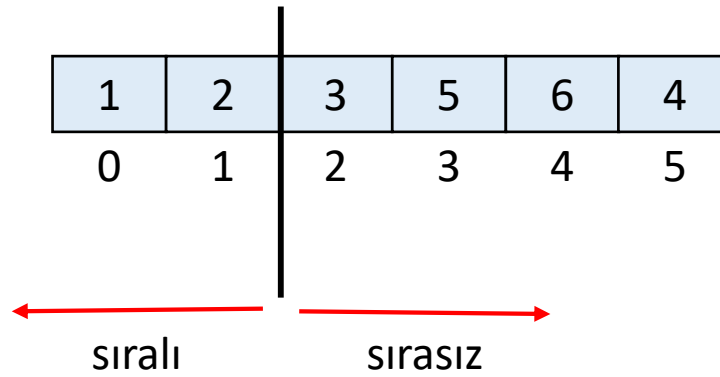
$\text{uzunluk} = 6$

Seçmeli Sıralama



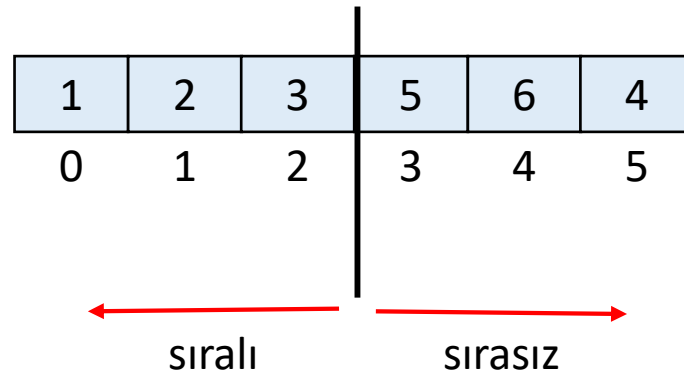
3. Tur

`min = 3`



`uzunluk = 6`

Seçmeli Sıralama

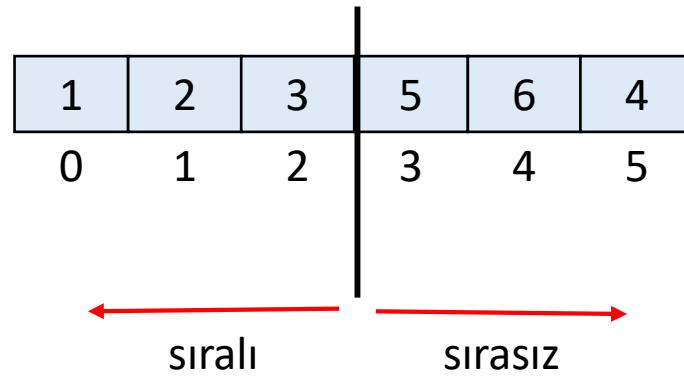


uzunluk = 6

Seçmeli Sıralama



4. Tur



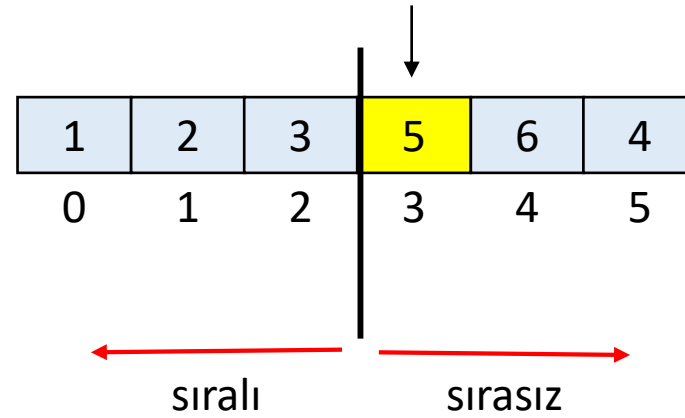
uzunluk = 6

Seçmeli Sıralama



4. Tur

min = 5



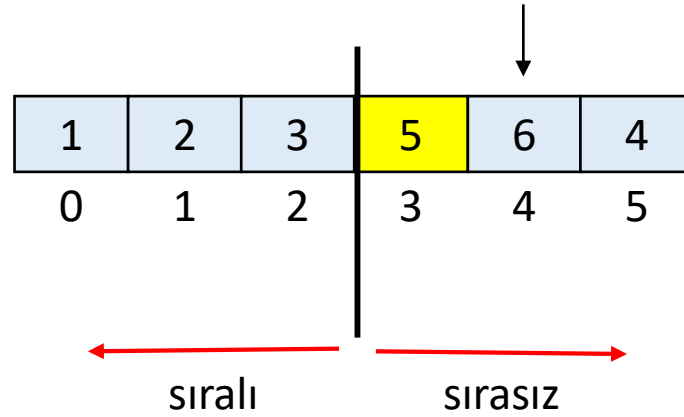
uzunluk = 6

Seçmeli Sıralama



4. Tur

$\text{min} = 5$
 $6 < \text{min} ?$



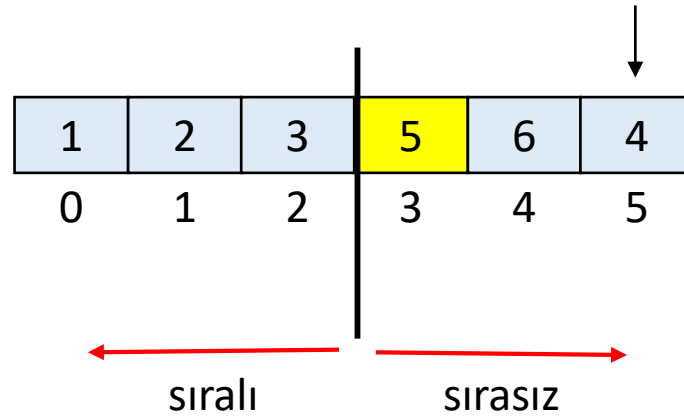
uzunluk = 6

Seçmeli Sıralama



4. Tur

$\text{min} = 5$
 $4 < \text{min} ?$



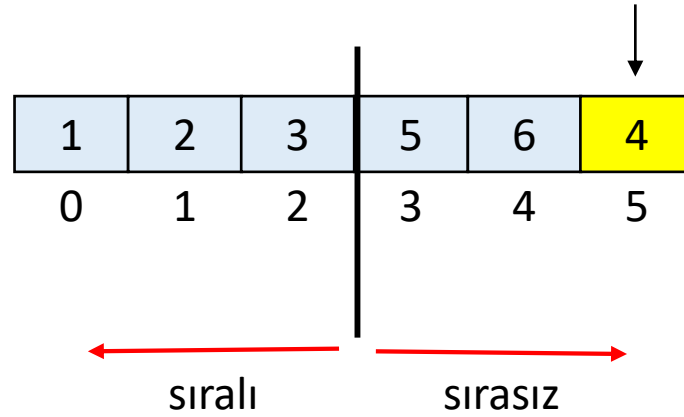
uzunluk = 6

Seçmeli Sıralama



4. Tur

$\text{min} = 4$



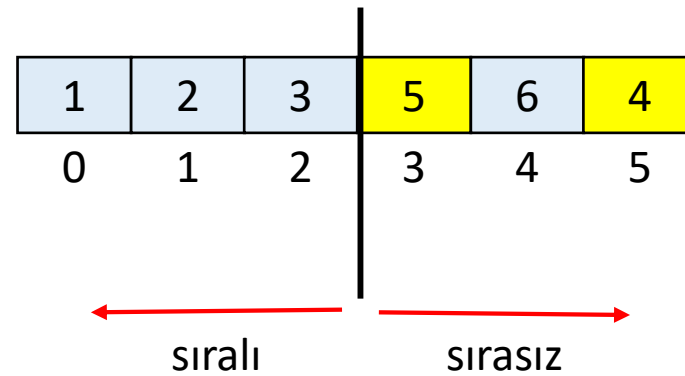
$\text{uzunluk} = 6$

Seçmeli Sıralama



4. Tur

$\text{min} = 4$



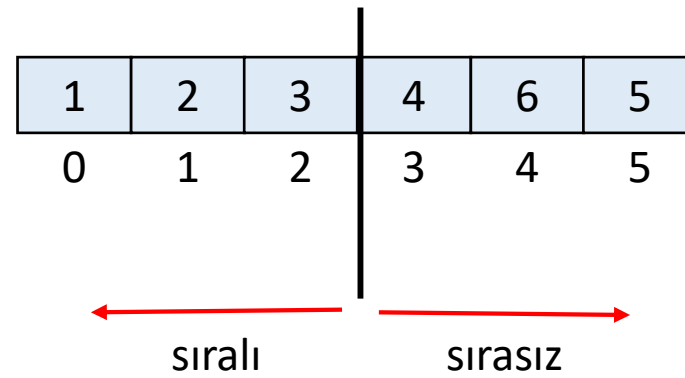
$\text{uzunluk} = 6$

Seçmeli Sıralama



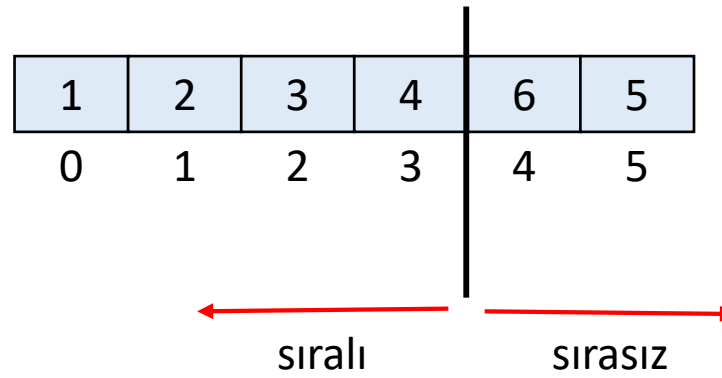
4. Tur

$\text{min} = 4$



$\text{uzunluk} = 6$

Seçmeli Sıralama



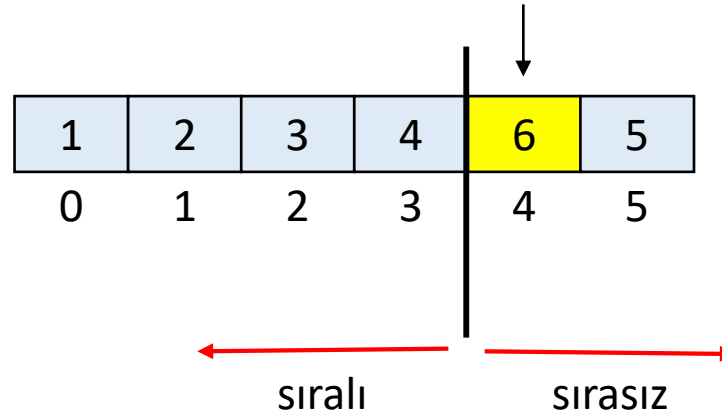
uzunluk = 6

Seçmeli Sıralama



5. Tur

min = 6



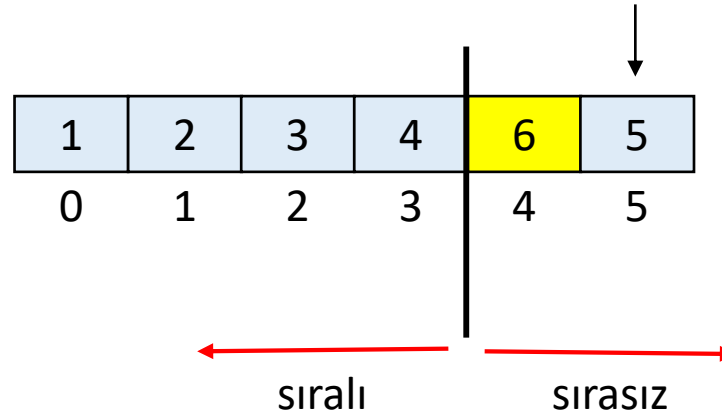
uzunluk = 6

Seçmeli Sıralama



5. Tur

$\text{min} = 6$
 $5 < \text{min} ?$



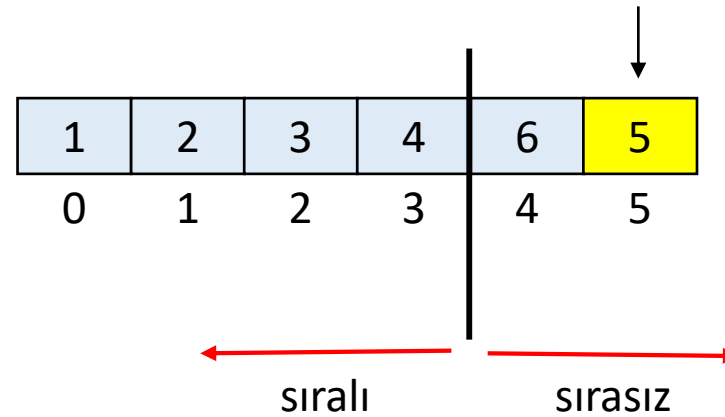
uzunluk = 6

Seçmeli Sıralama



5. Tur

$\text{min} = 5$



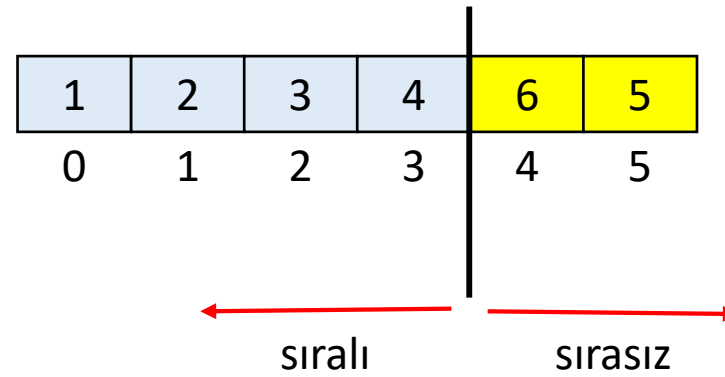
$\text{uzunluk} = 6$

Seçmeli Sıralama



5. Tur

$\text{min} = 5$



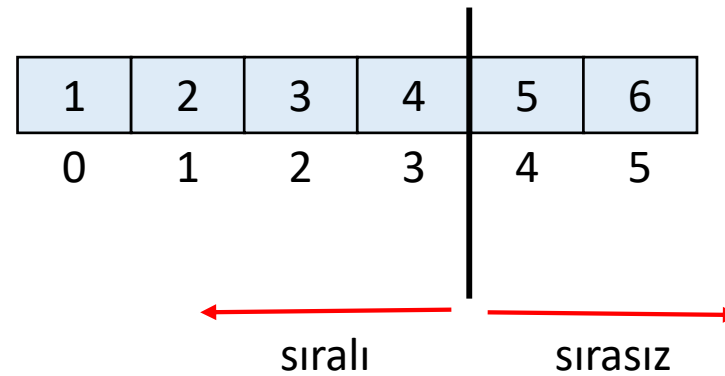
$\text{uzunluk} = 6$

Seçmeli Sıralama



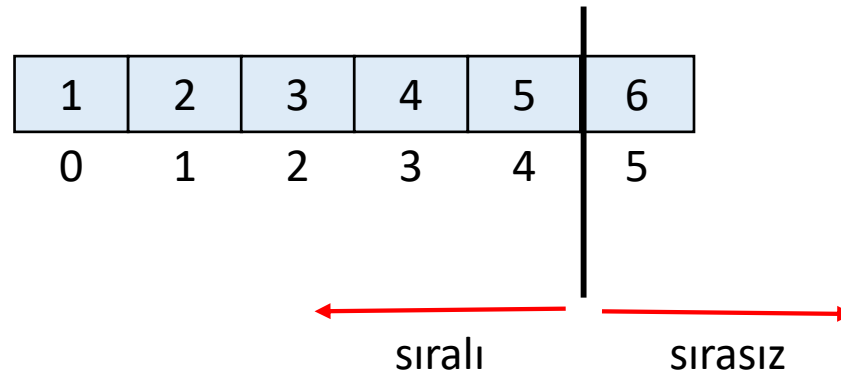
5. Tur

$\text{min} = 5$



$\text{uzunluk} = 6$

Seçmeli Sıralama



uzunluk = 6

Seçmeli Sıralama



1	2	3	4	5	6
0	1	2	3	4	5

uzunluk = 6

Seçmeli Sıralama



```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama



dizi[]	5	1	10	2	9
	0	1	2	3	4

```
→ public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama



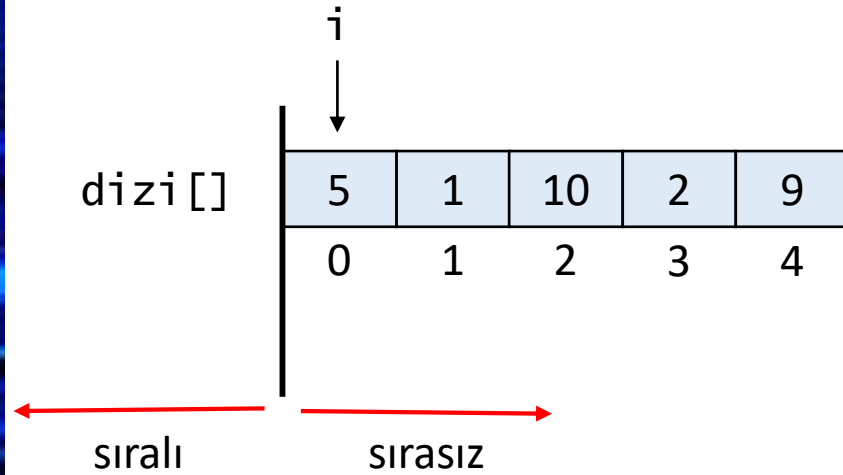
dizi[]	5	1	10	2	9
	0	1	2	3	4

n = 5



```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama



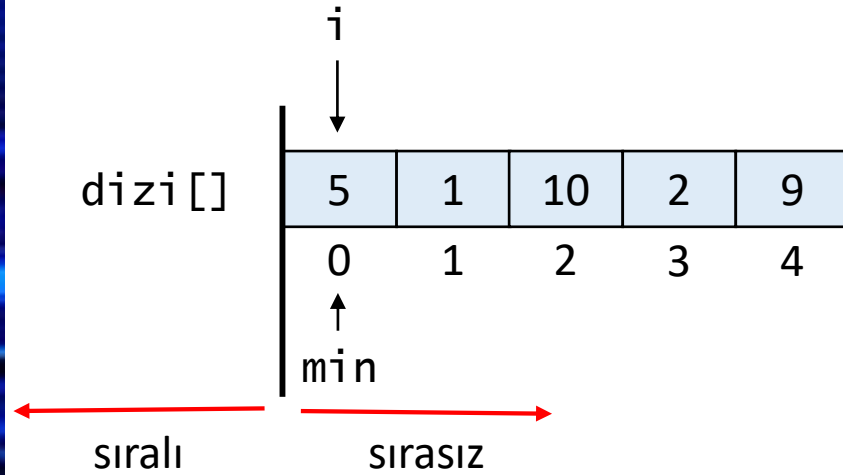
`i = 0`

`n = 5`



```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama



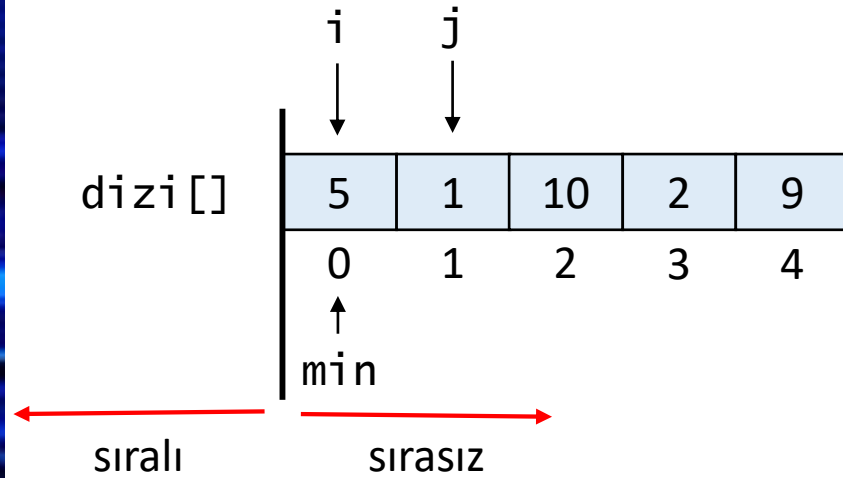
$i = 0$
 $min = 0$

$n = 5$



```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

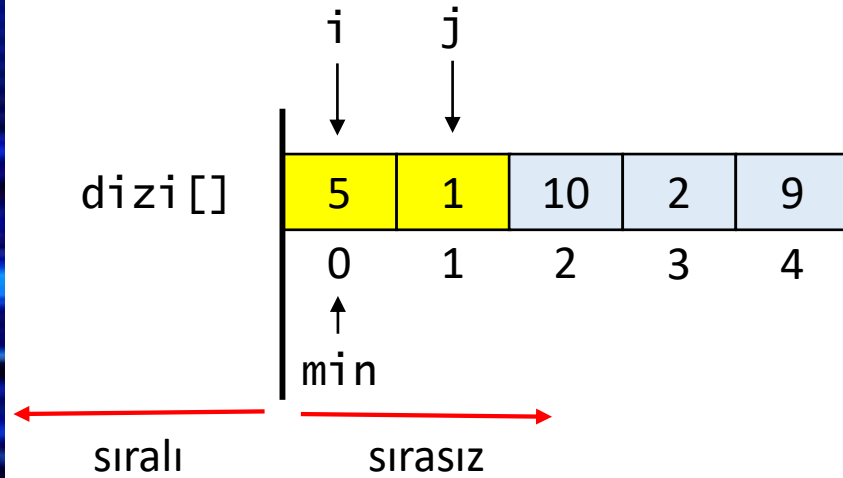


`i = 0`
`min = 0`
`j = 1`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

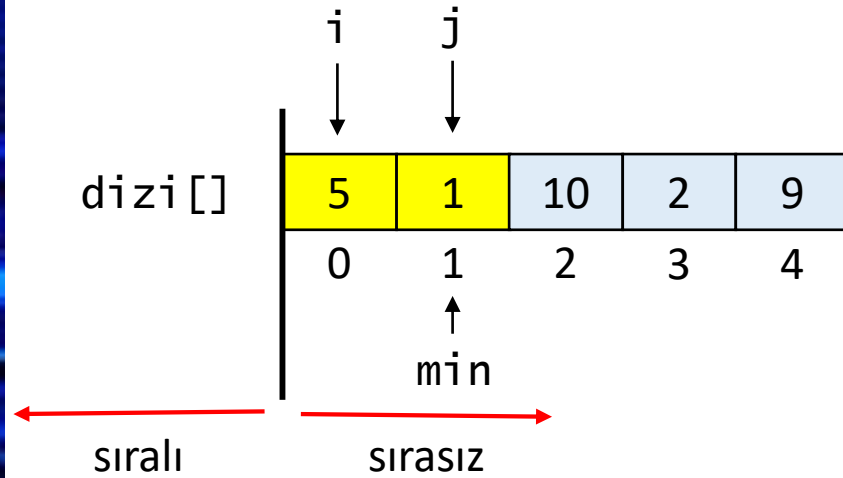


`i = 0`
`min = 0`
`j = 1`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```


Seçmeli Sıralama

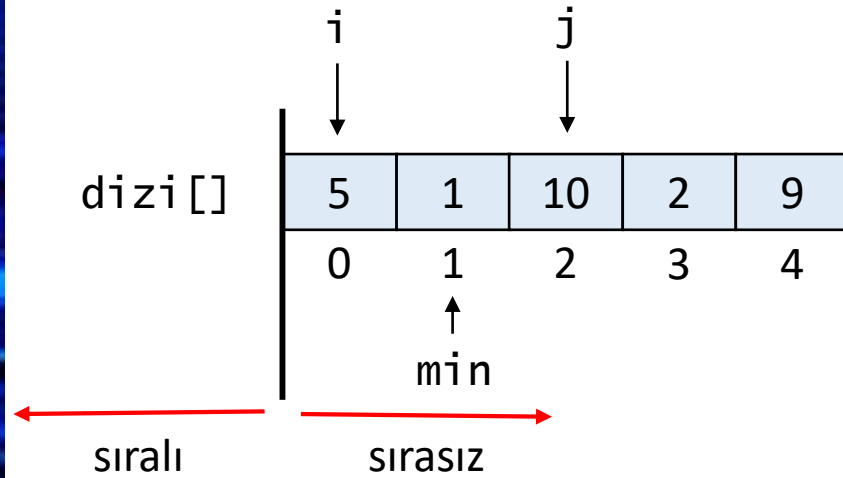


`i = 0`
`min = 1`
`j = 1`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

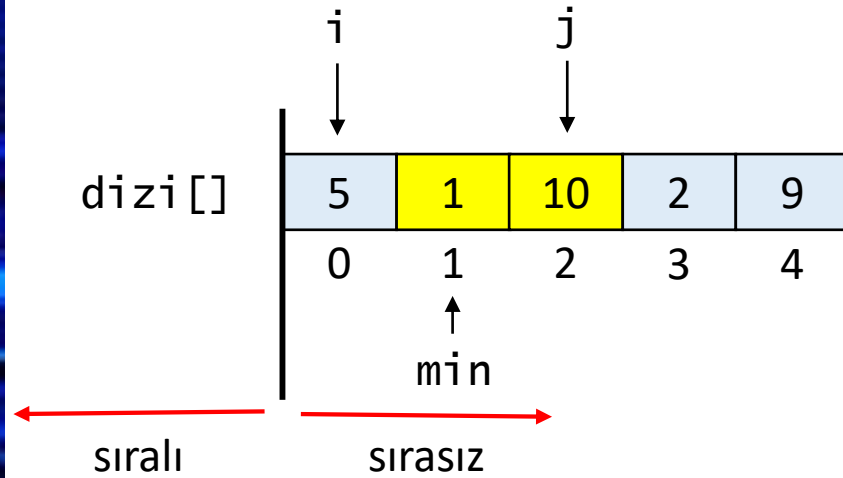


$i = 0$
 $\text{min} = 1$
 $j = 2$

$n = 5$

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

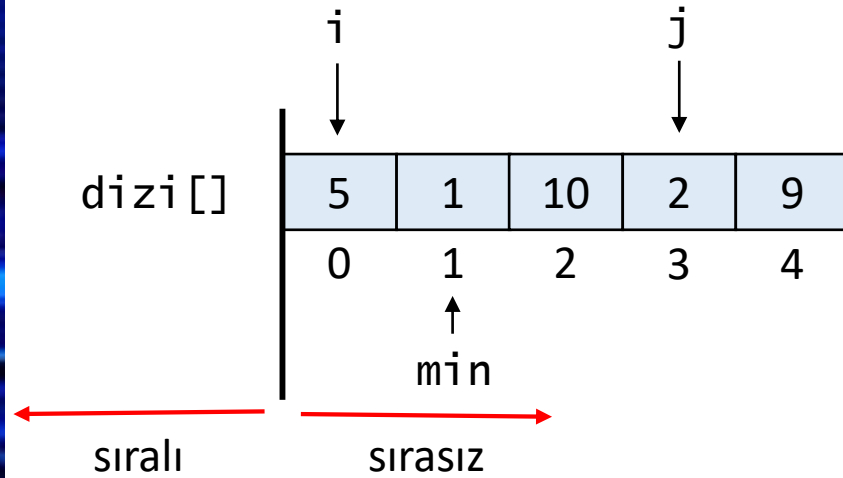


$i = 0$
 $\min = 1$
 $j = 2$

$n = 5$

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

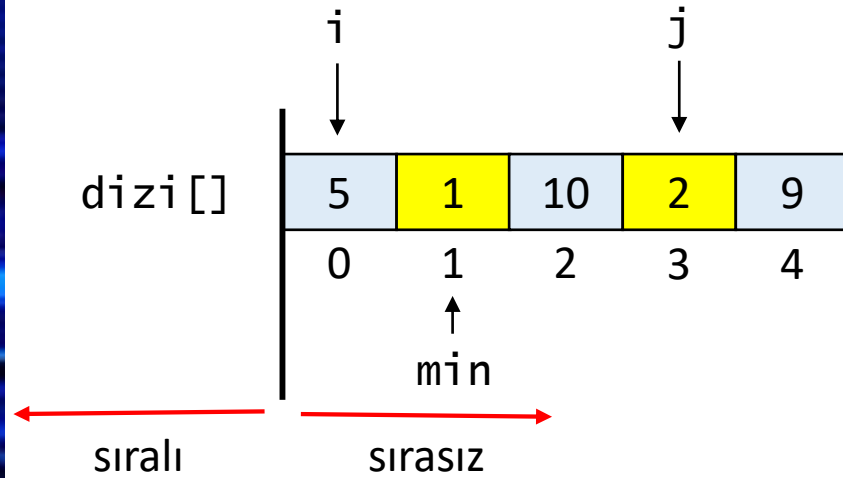


$i = 0$
 $min = 1$
 $j = 3$

$n = 5$

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

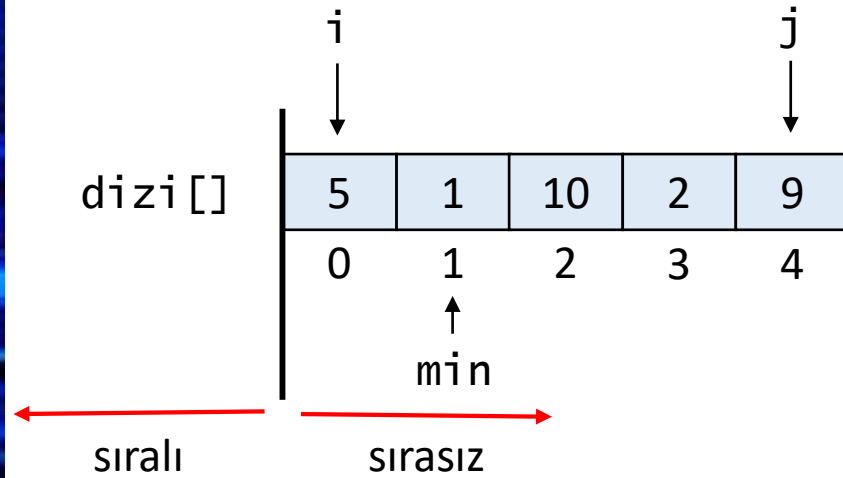


`i = 0`
`min = 1`
`j = 3`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

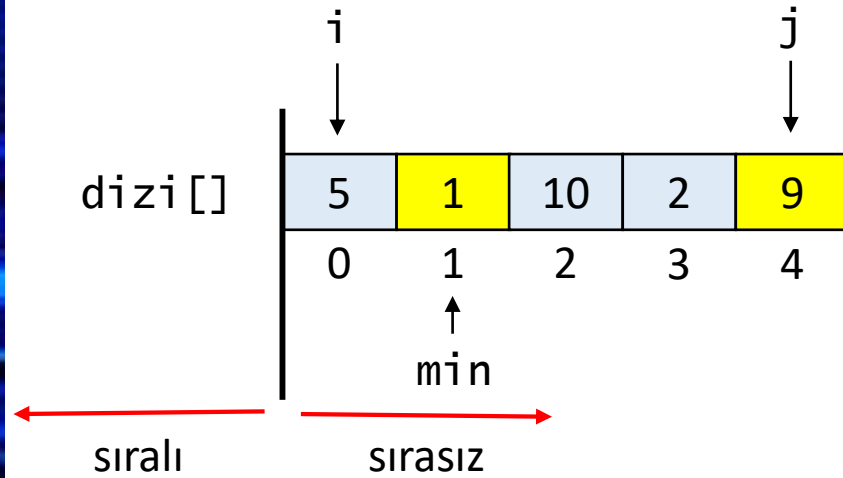


$i = 0$
 $\text{min} = 1$
 $j = 4$

$n = 5$

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

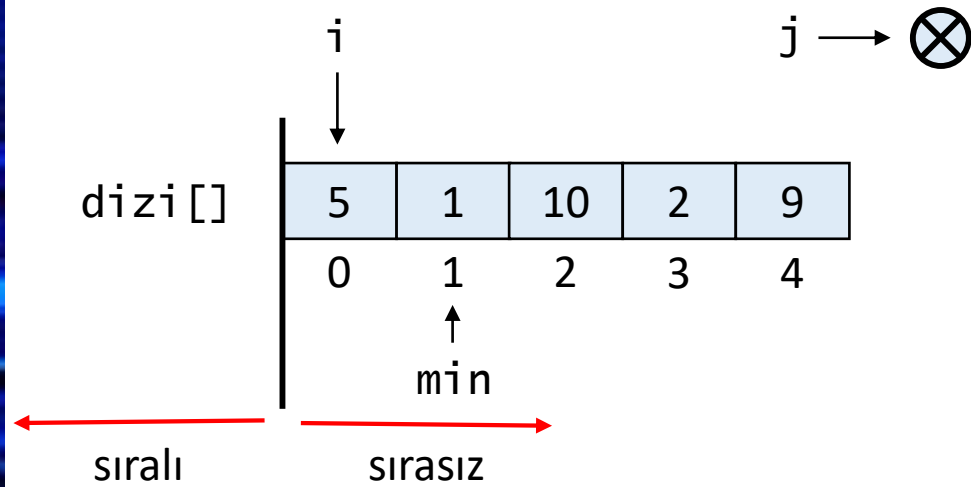


`i = 0`
`min = 1`
`j = 4`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

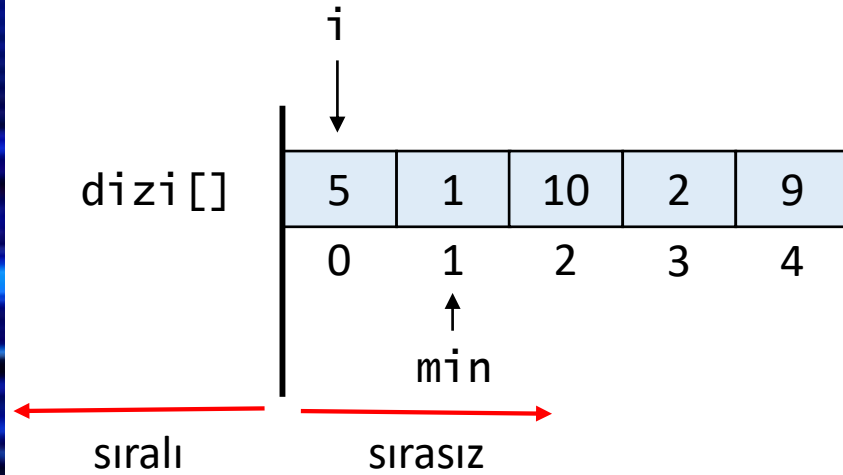


`i = 0`
`min = 1`
`j = 5`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```


Seçmeli Sıralama

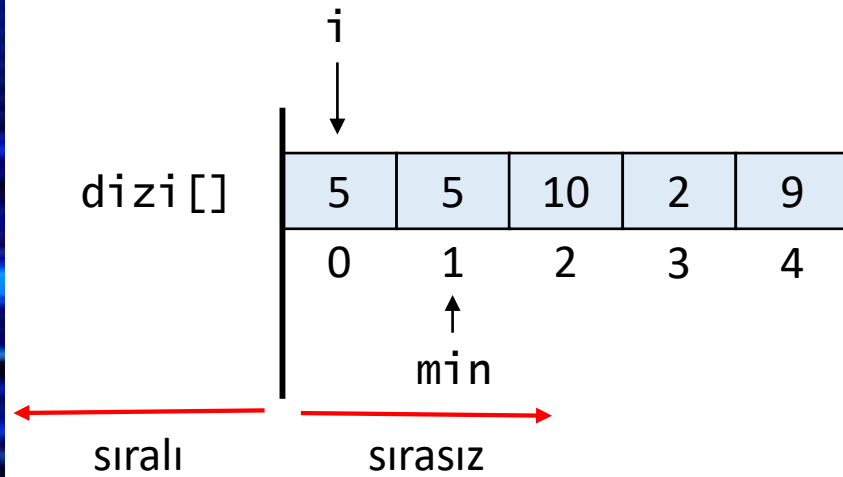


`i = 0`
`min = 1`
`gecici = 1`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

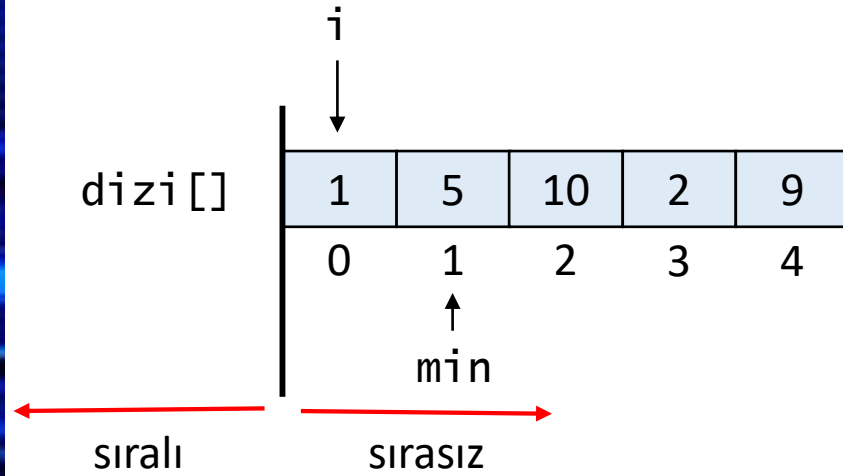


`i = 0`
`min = 1`
`gecici = 1`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

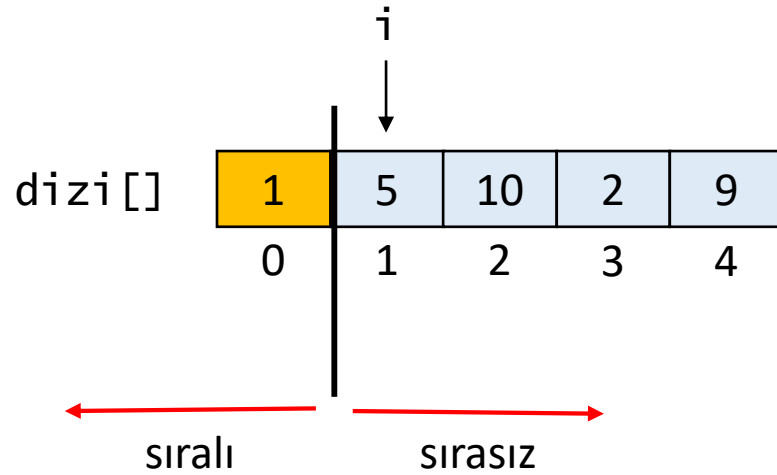


`i = 0`
`min = 1`
`gecici = 1`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama



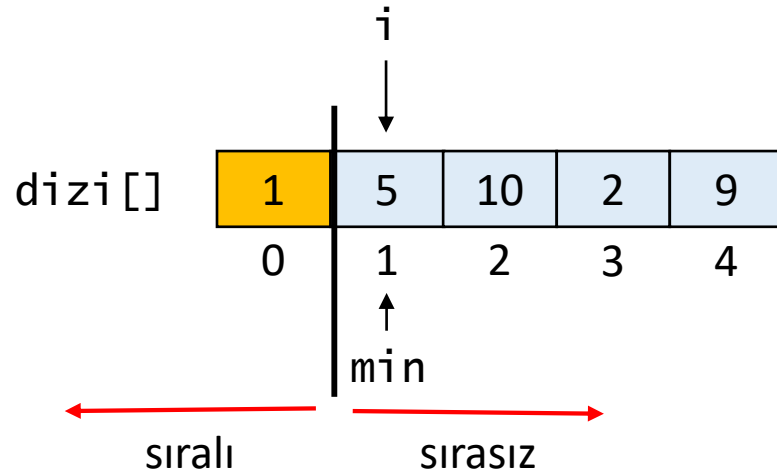
`i = 1`

`n = 5`



```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama



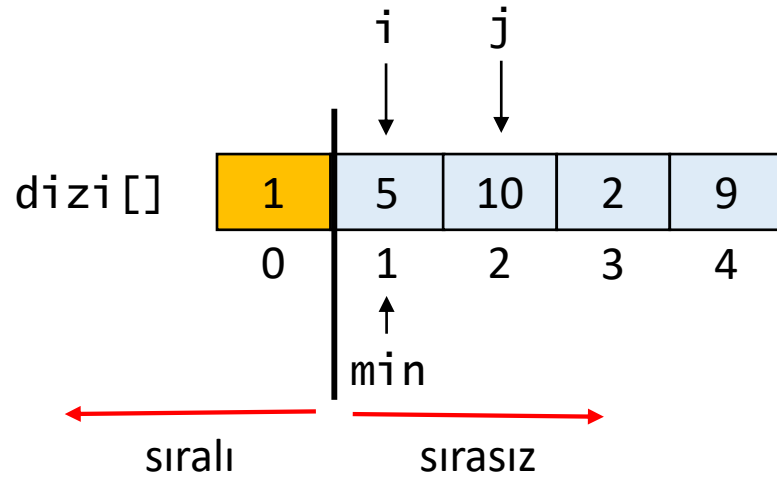
`i = 1`
`min = 1`

`n = 5`



```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

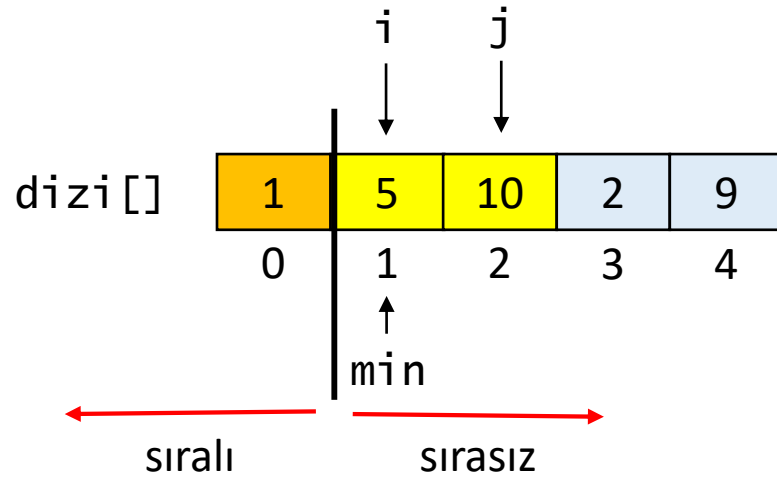


`i = 1`
`min = 1`
`j = 2`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

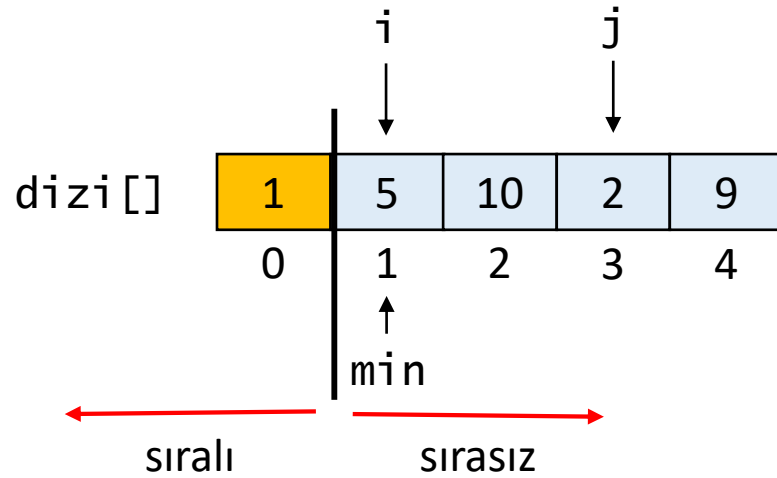


`i = 1`
`min = 1`
`j = 2`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

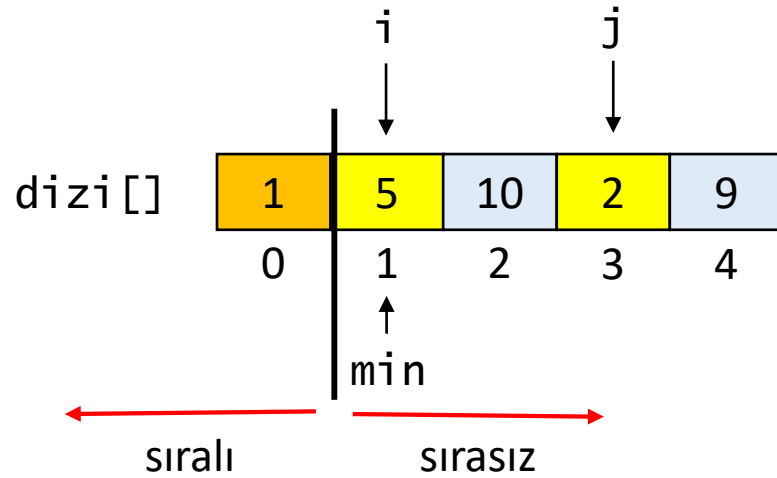


`i = 1`
`min = 1`
`j = 3`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```


Seçmeli Sıralama

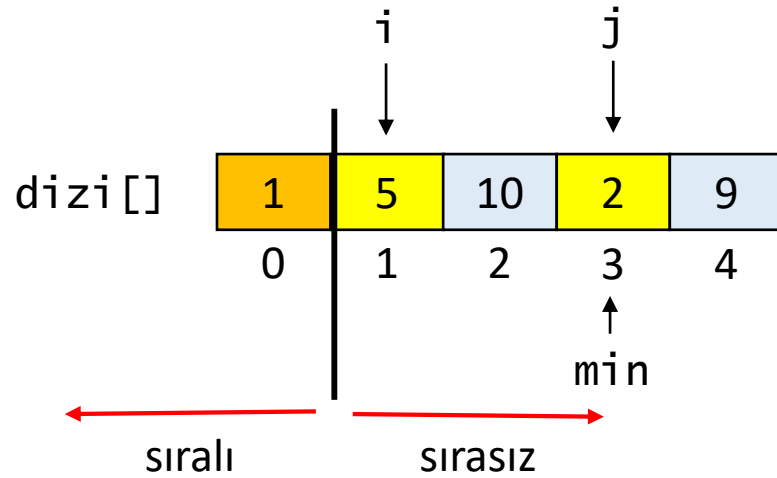


`i = 1`
`min = 1`
`j = 3`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

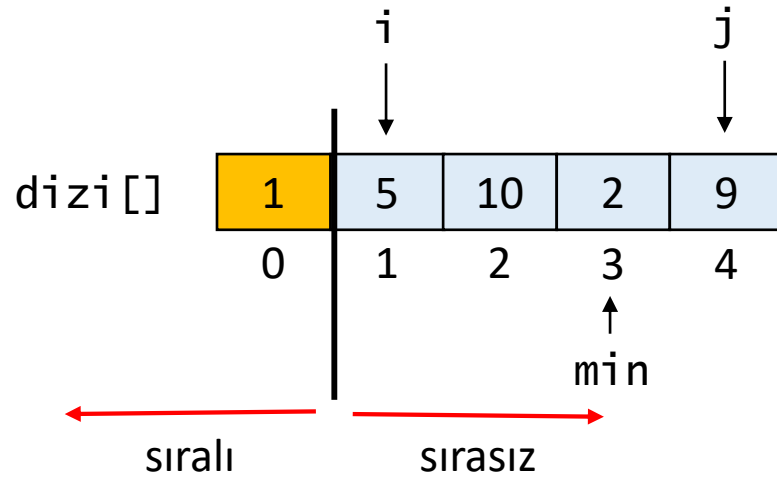


`i = 1`
`min = 3`
`j = 3`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

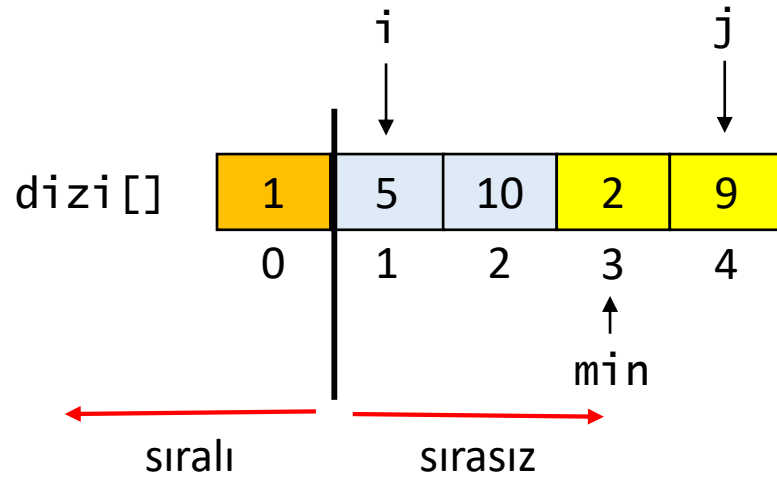


`i = 1`
`min = 3`
`j = 4`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

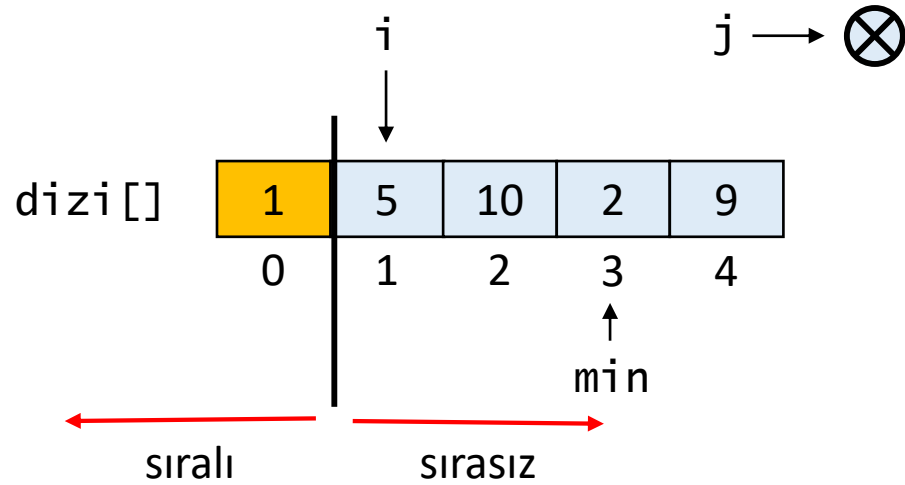


`i = 1`
`min = 3`
`j = 4`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

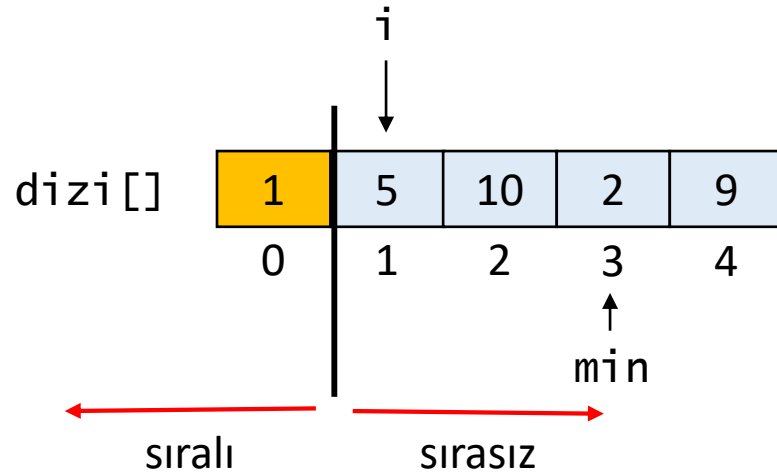


`i = 1`
`min = 3`
`j = 5`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

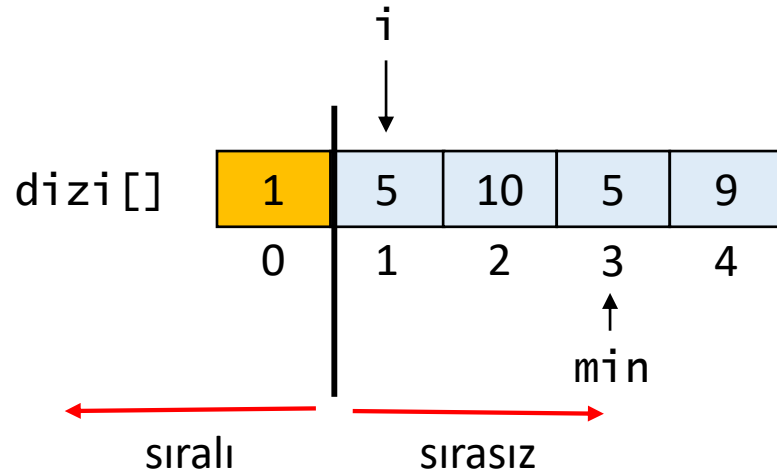


`i = 1`
`min = 3`
`gecici = 2`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

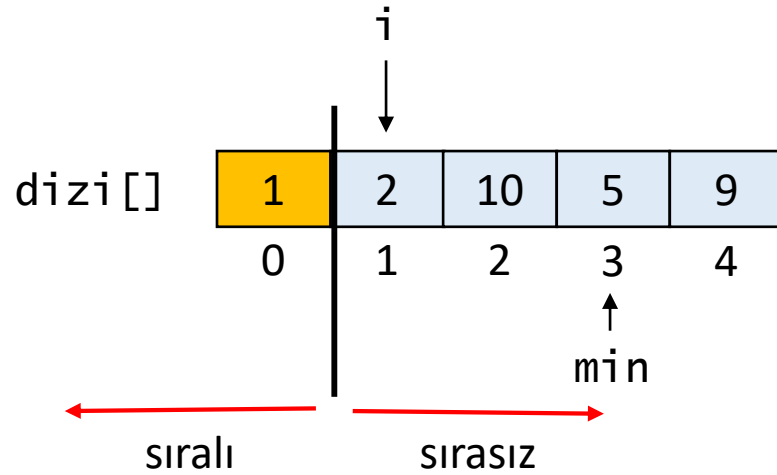


`i = 1`
`min = 3`
`gecici = 2`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

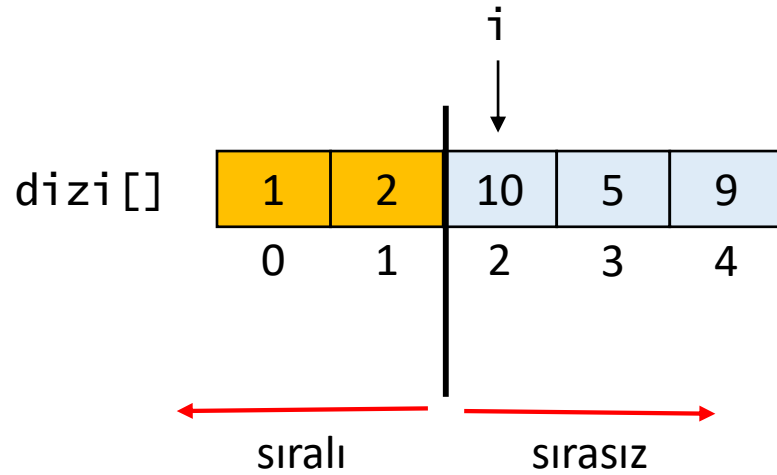


`i = 1`
`min = 3`
`gecici = 2`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```


Seçmeli Sıralama



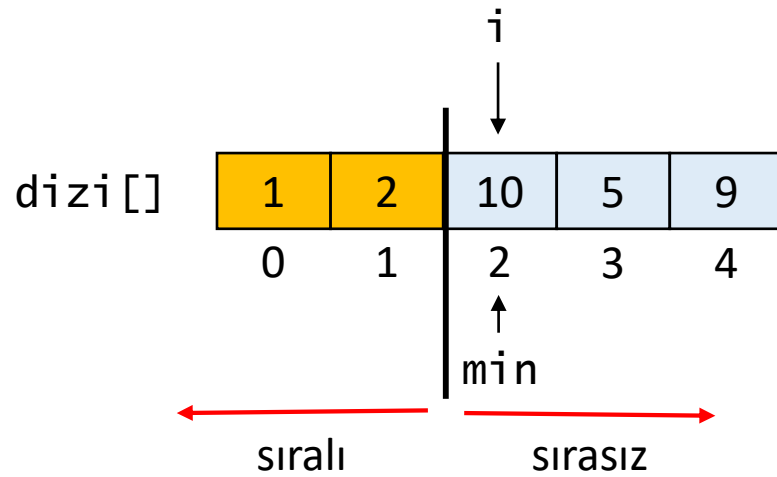
`i = 2`

`n = 5`



```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

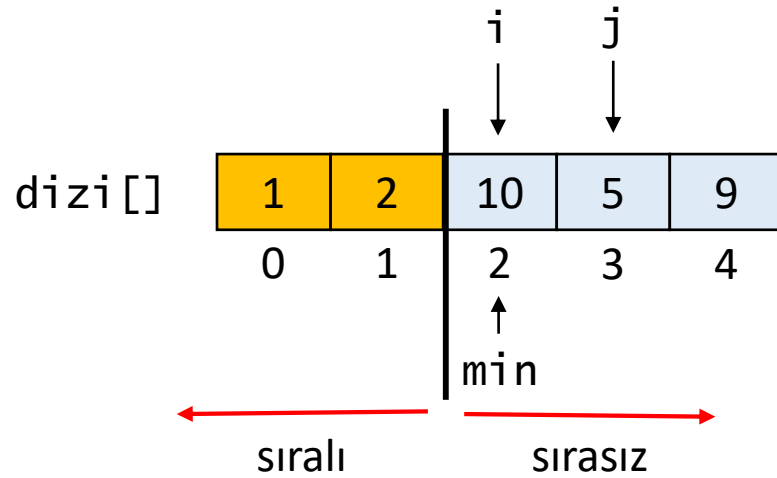


`i = 2`
`min = 2`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

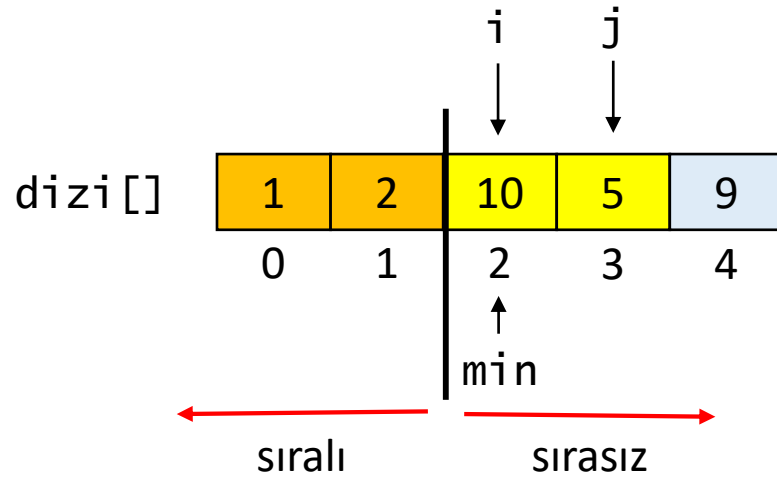


`i = 2`
`min = 2`
`j = 3`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

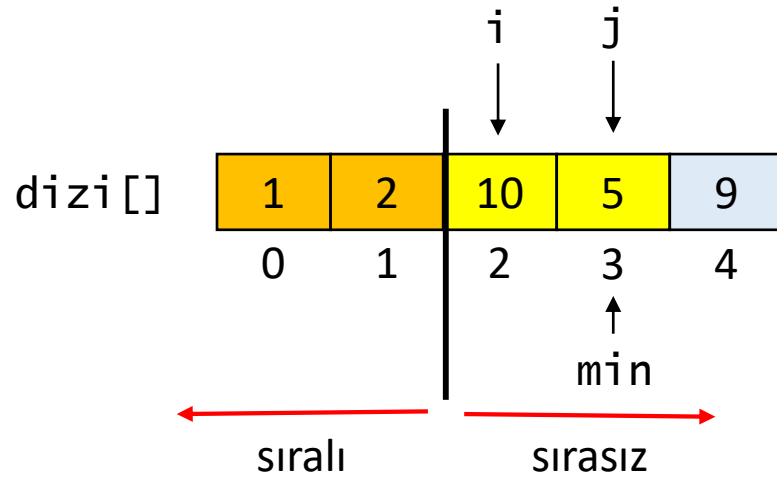


`i = 2`
`min = 2`
`j = 3`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

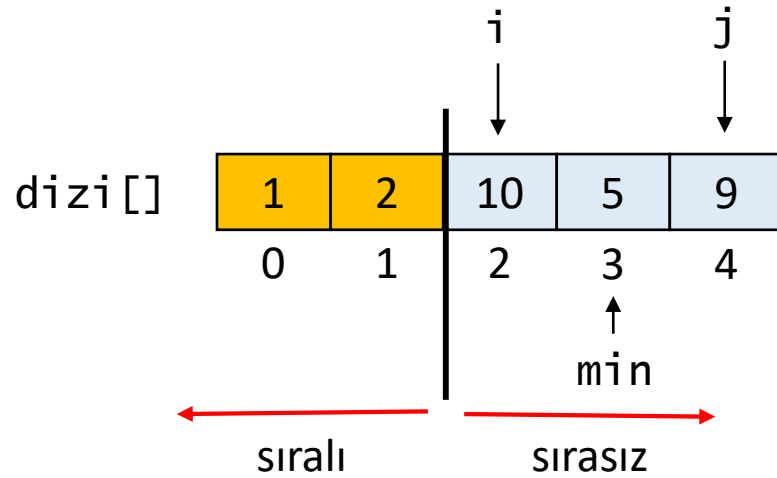


$i = 2$
 $min = 3$
 $j = 3$

$n = 5$

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

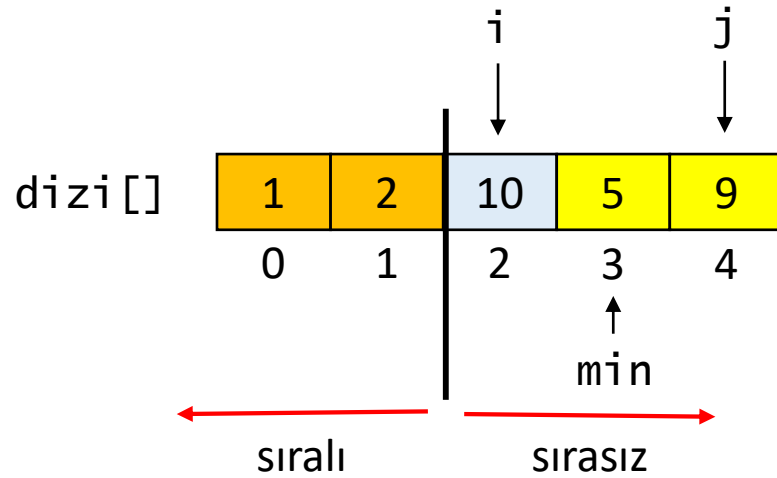


`i = 2`
`min = 3`
`j = 4`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

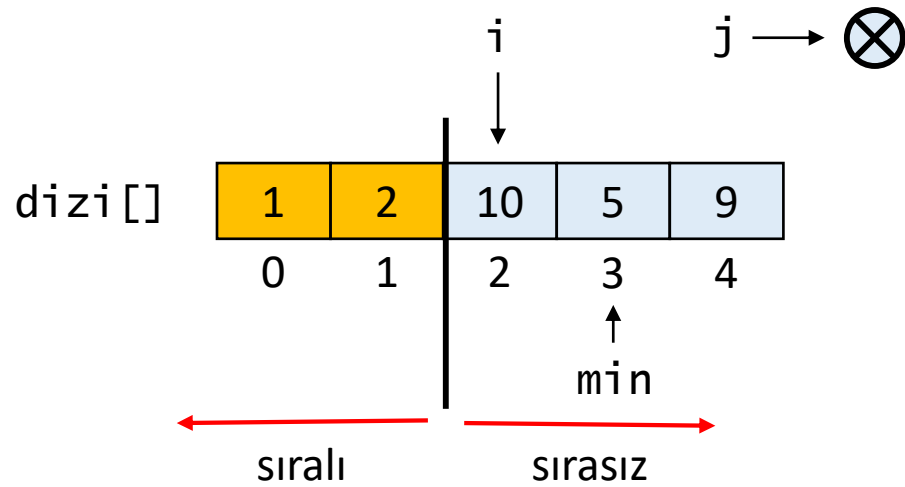


`i = 2`
`min = 3`
`j = 4`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

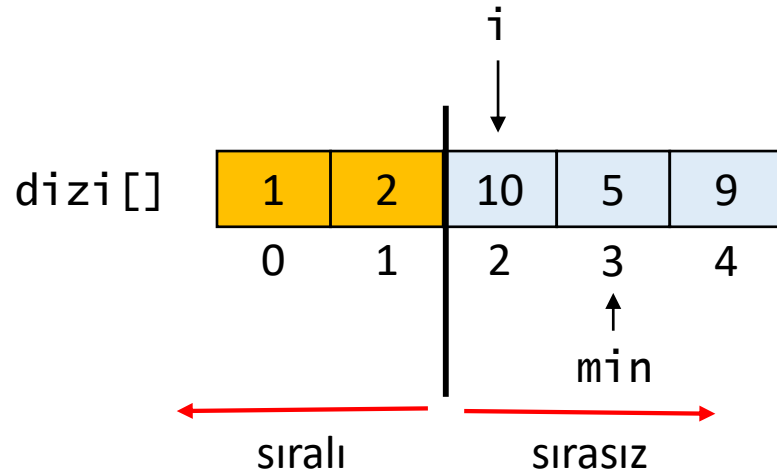


`i = 2`
`min = 3`
`j = 5`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```


Seçmeli Sıralama

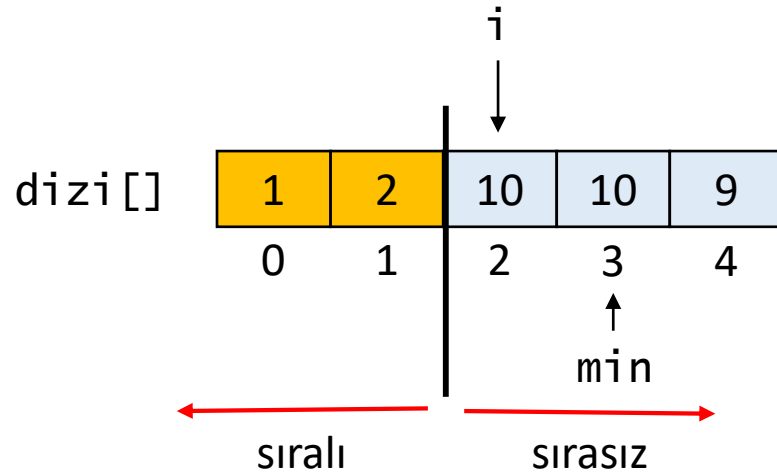


`i = 2`
`min = 3`
`gecici = 5`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

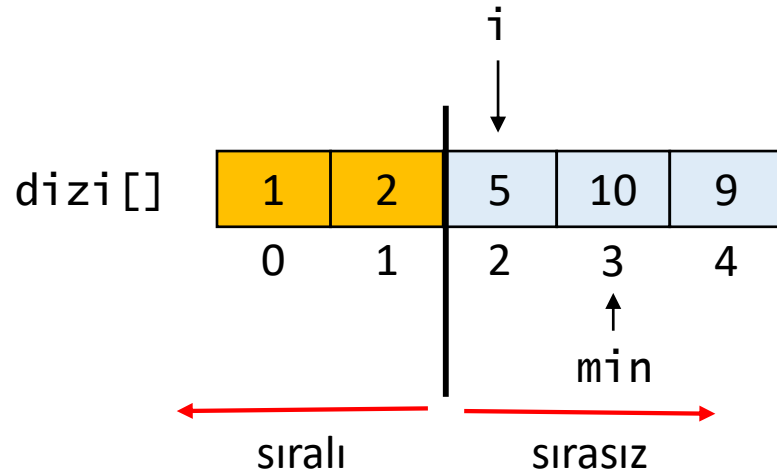


`i = 2`
`min = 3`
`gecici = 5`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

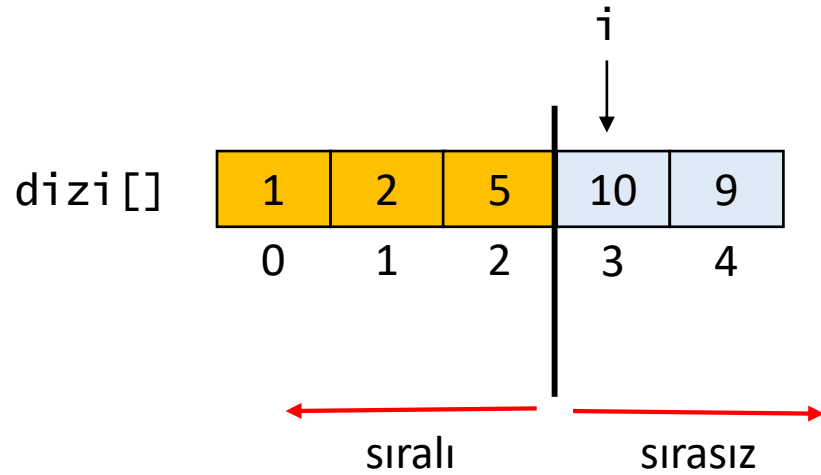


`i = 2`
`min = 3`
`gecici = 5`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama



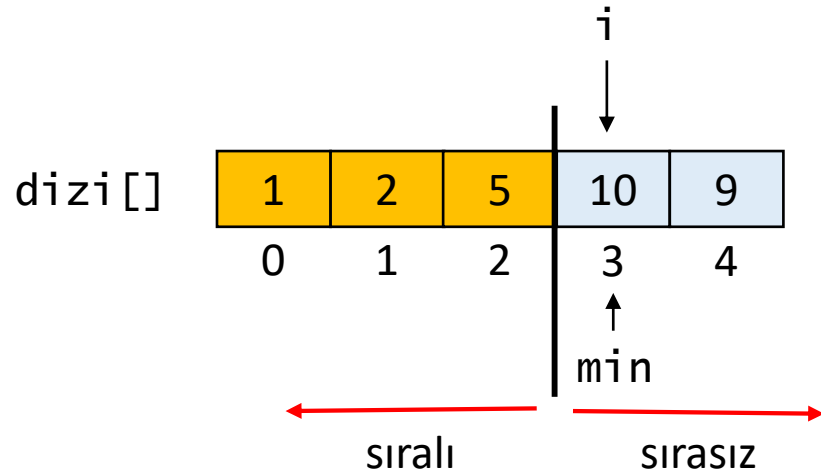
i = 3

n = 5



```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

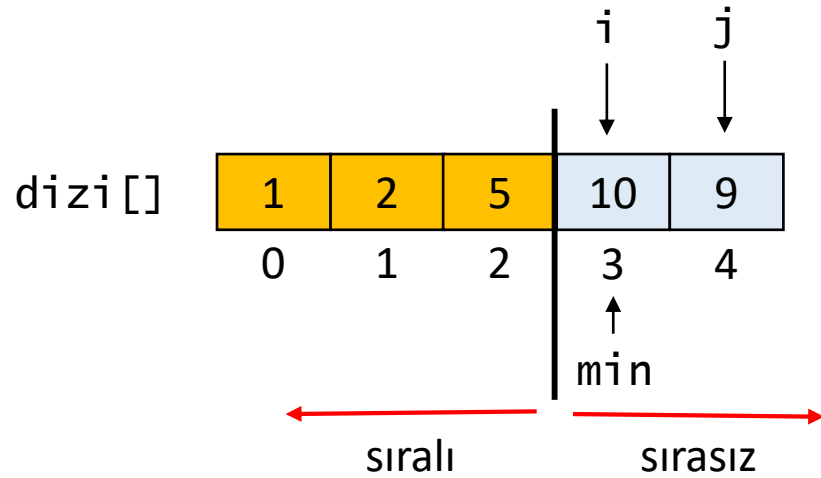


i = 3
min = 3

n = 5

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

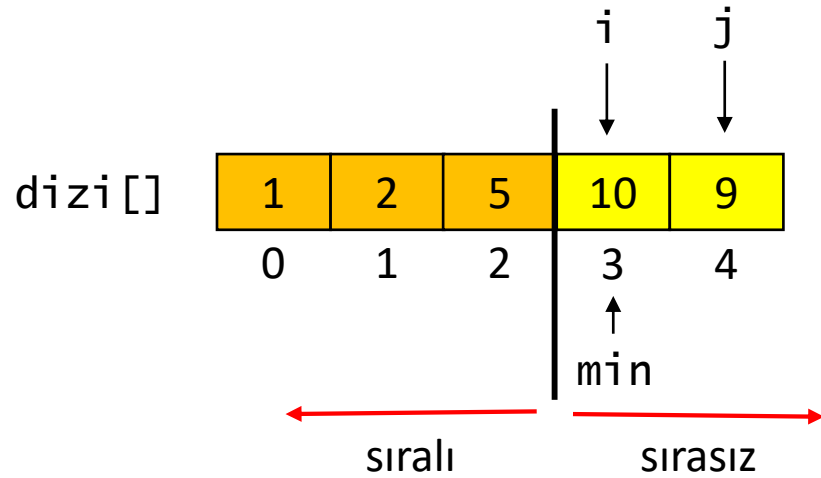


i = 3
min = 3
j = 4

n = 5

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

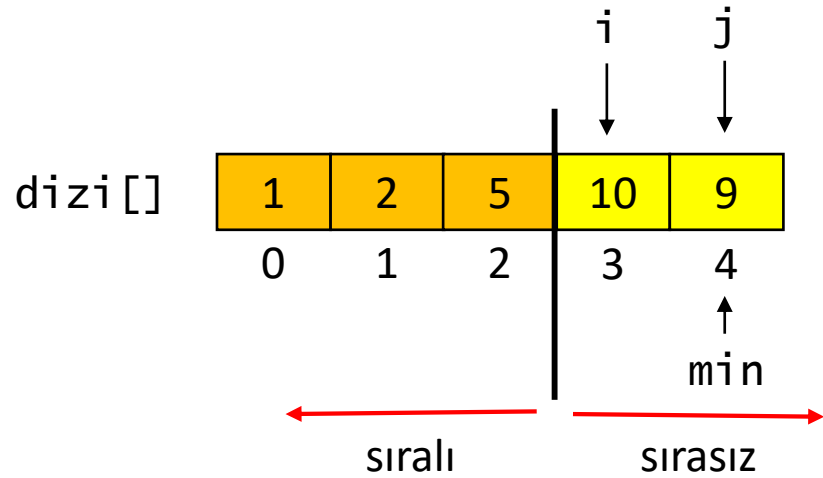


i = 3
min = 3
j = 4

n = 5

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

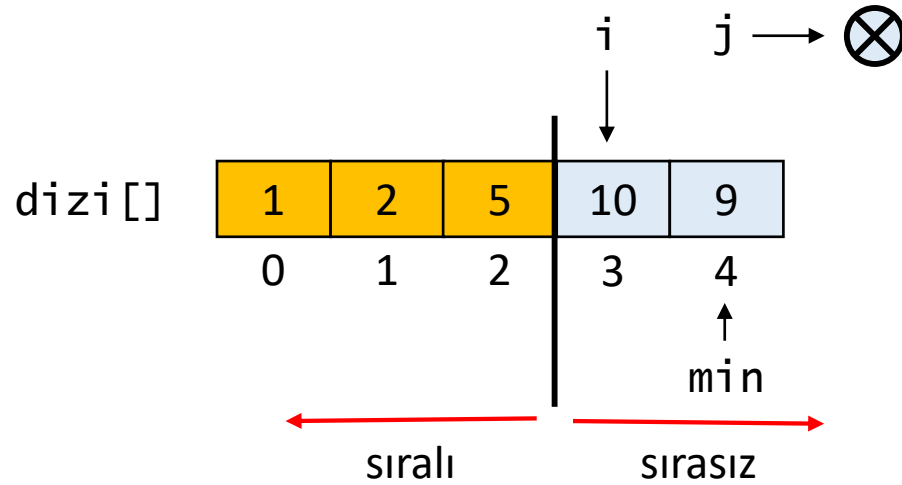


i = 3
min = 4
j = 4

n = 5

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```


Seçmeli Sıralama

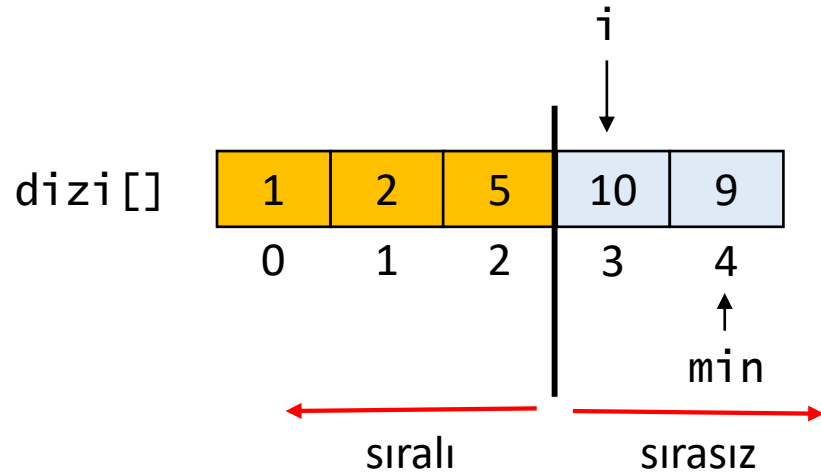


`i = 3`
`min = 4`
`j = 5`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

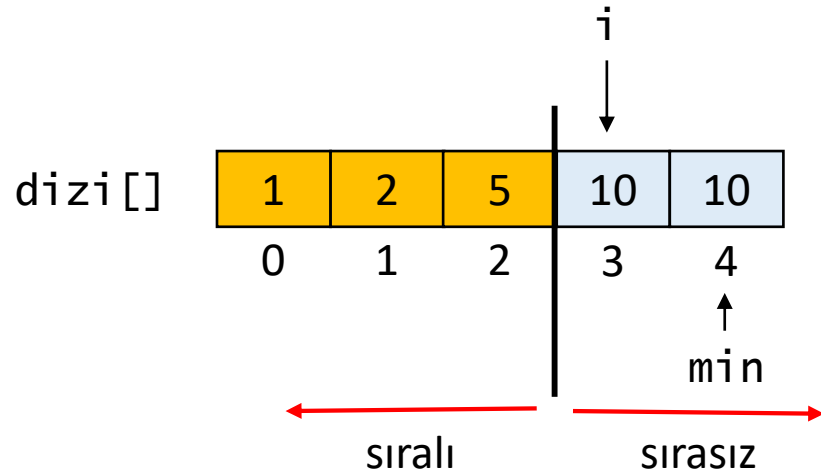


`i = 3`
`min = 4`
`gecici = 9`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

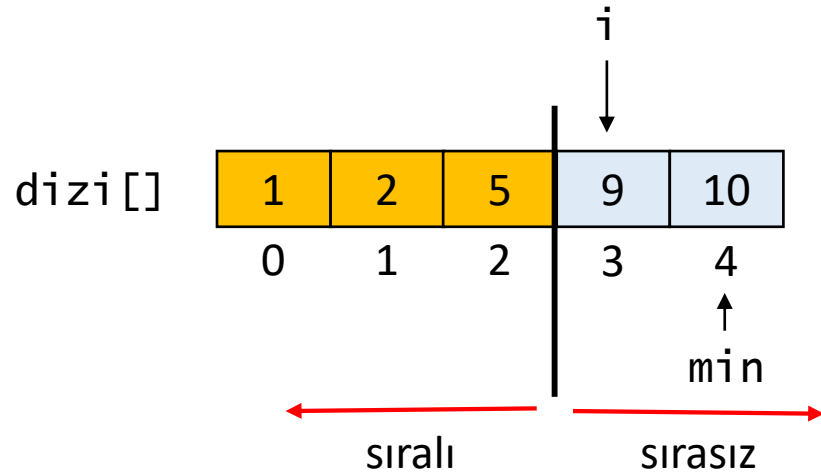


`i = 3`
`min = 4`
`gecici = 9`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama

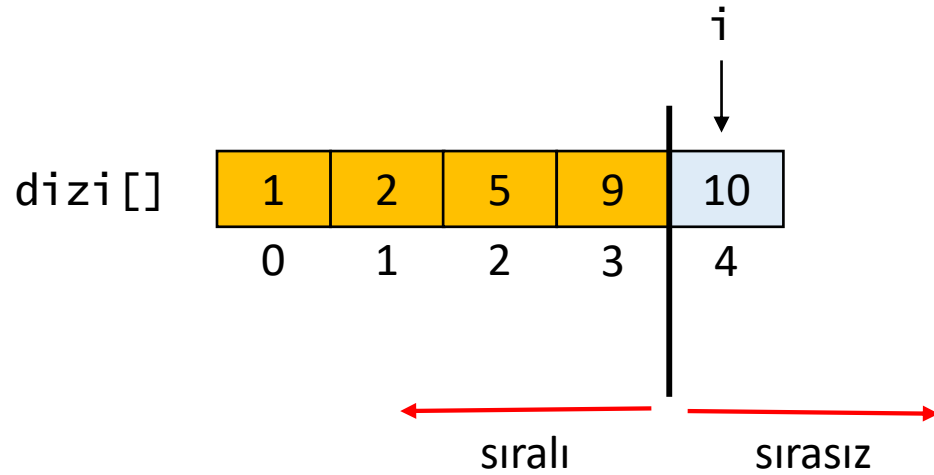


`i = 3`
`min = 4`
`gecici = 9`

`n = 5`

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama



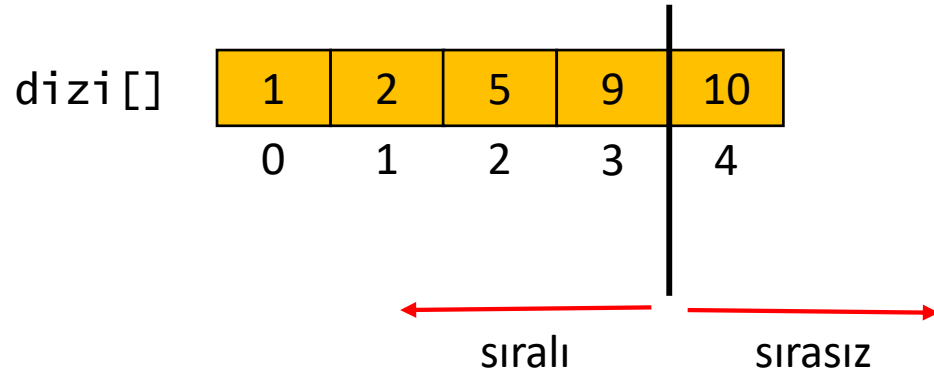
i = 4

n = 5



```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama



n = 5

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```

Seçmeli Sıralama



dizi[]	1	2	5	9	10
	0	1	2	3	4

```
public void sort(int[] dizi) {  
    int n = dizi.length;  
    for(int i = 0; i < n - 1; i++) {  
        int min = i;  
        for(int j = i + 1; j < n; j++) {  
            if(dizi[j] < dizi[min]) {  
                min = j;  
            }  
        }  
        int gecici = dizi[min];  
        dizi[min] = dizi[i];  
        dizi[i] = gecici;  
    }  
}
```



Çabuk Sıralama (QuickSort)

- Bir pivot nokta seç.
- Pivot noktadan diziyi iki parçaya böl.
- İki alt parçayı özyinelemeli olarak sırala.



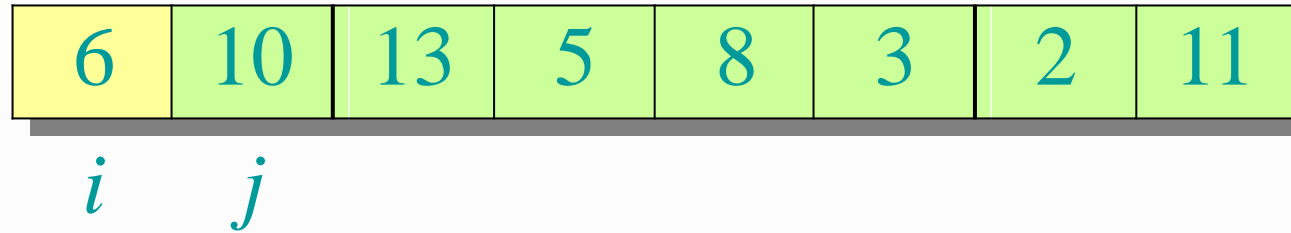


Parçalama (Partitioning) Prosedürü

```
function partition(A, p, q):  
    pivot = A[q]  // pivot seç  
    i = p - 1      // en küçük eleman indisi  
    for j from p to q - 1:  
        if A[j] <= pivot:  
            i = i + 1  
            swap(A[i], A[j])  
    swap(A[i + 1], A[q]) // pivot ve elemanı yer deęiş  
    return i + 1 // parçalamadan sonra pivotun indisi
```

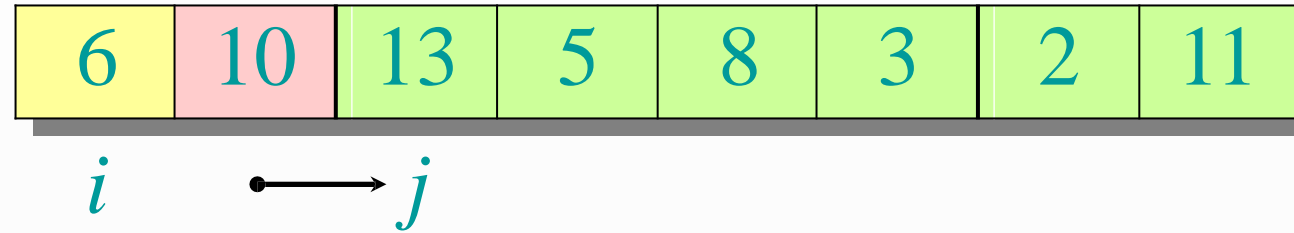


Parçalara Ayırma (Partitioning)



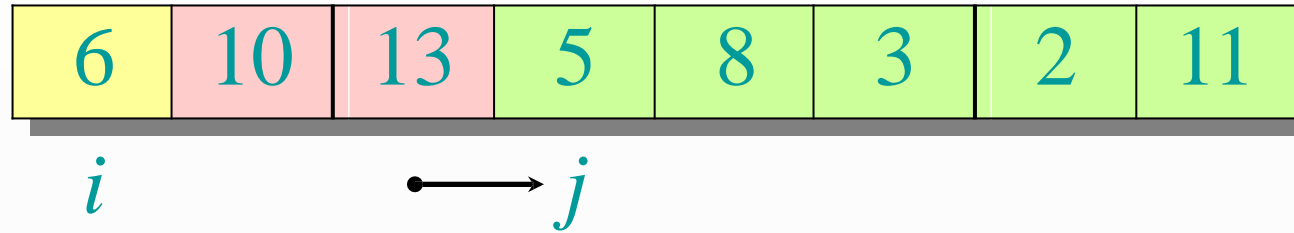


Parçalara Ayırma (Partitioning)



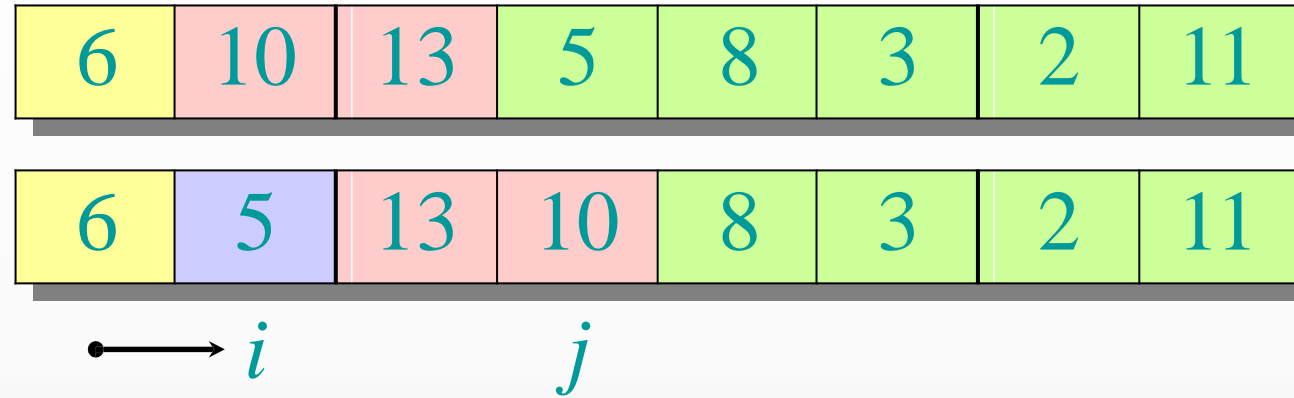


Parçalara Ayırma (Partitioning)



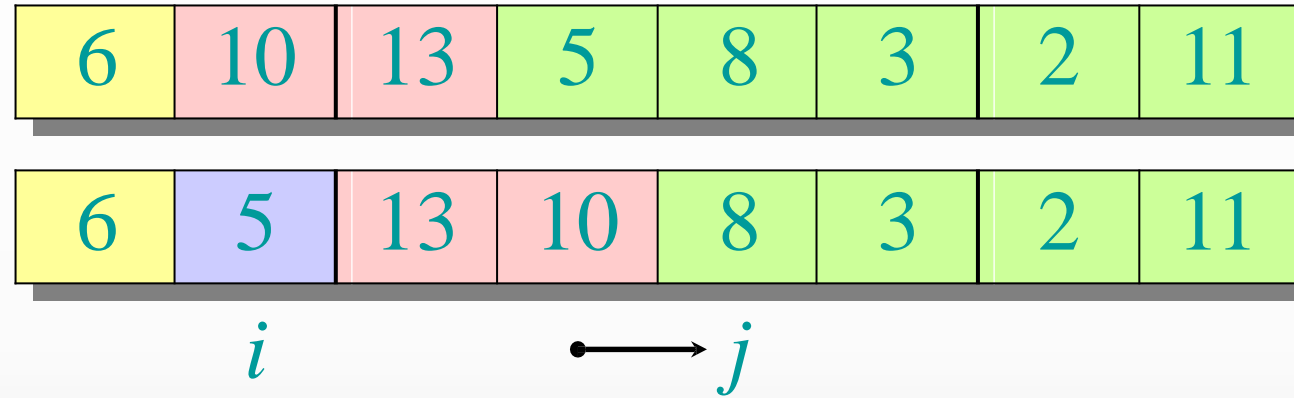


Parçalara Ayırma (Partitioning)



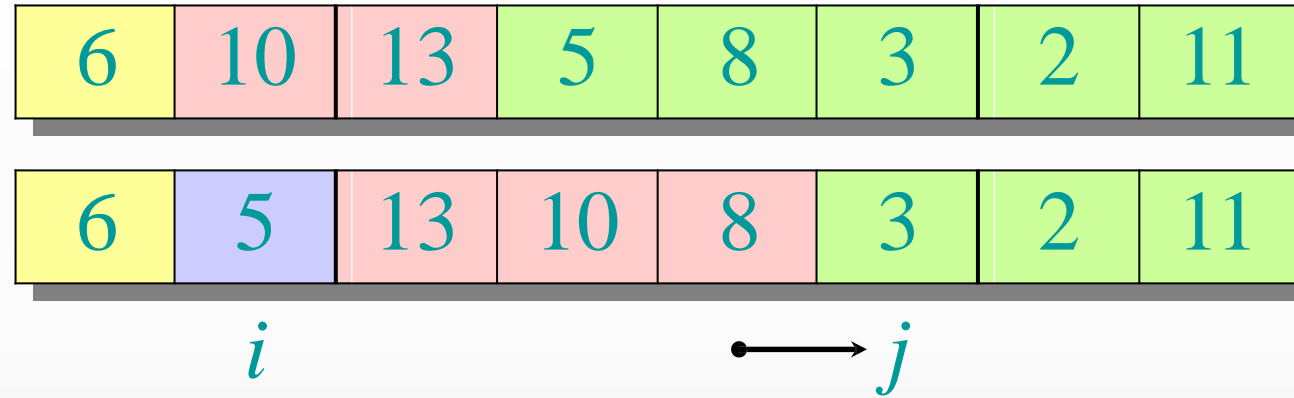


Parçalara Ayırma (Partitioning)



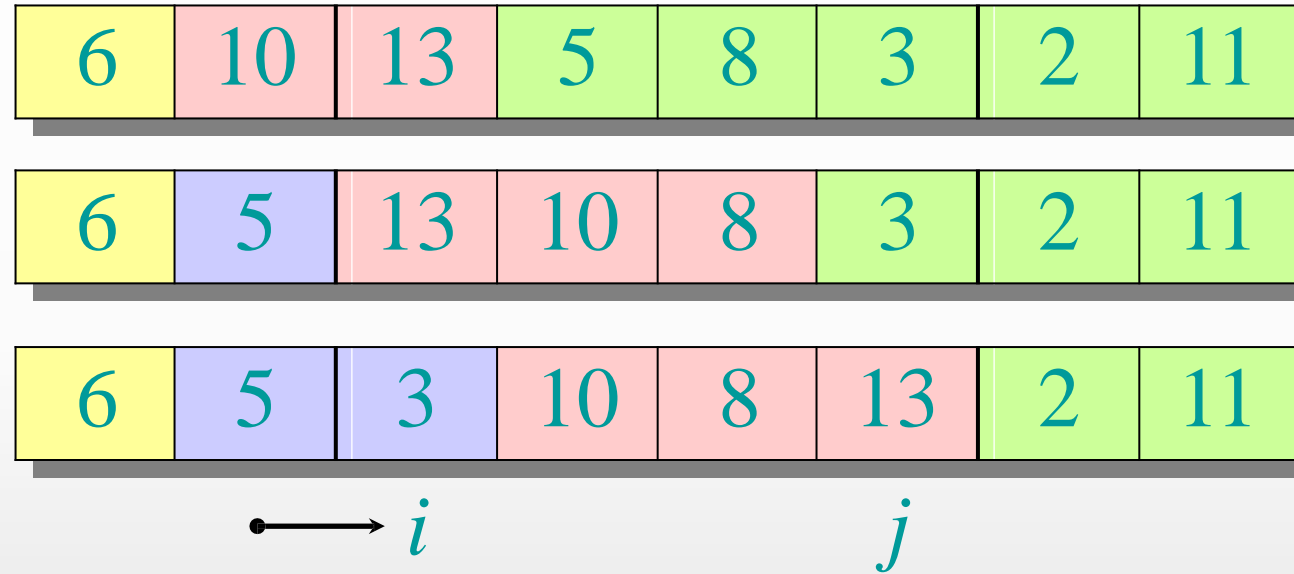


Parçalara Ayırma (Partitioning)



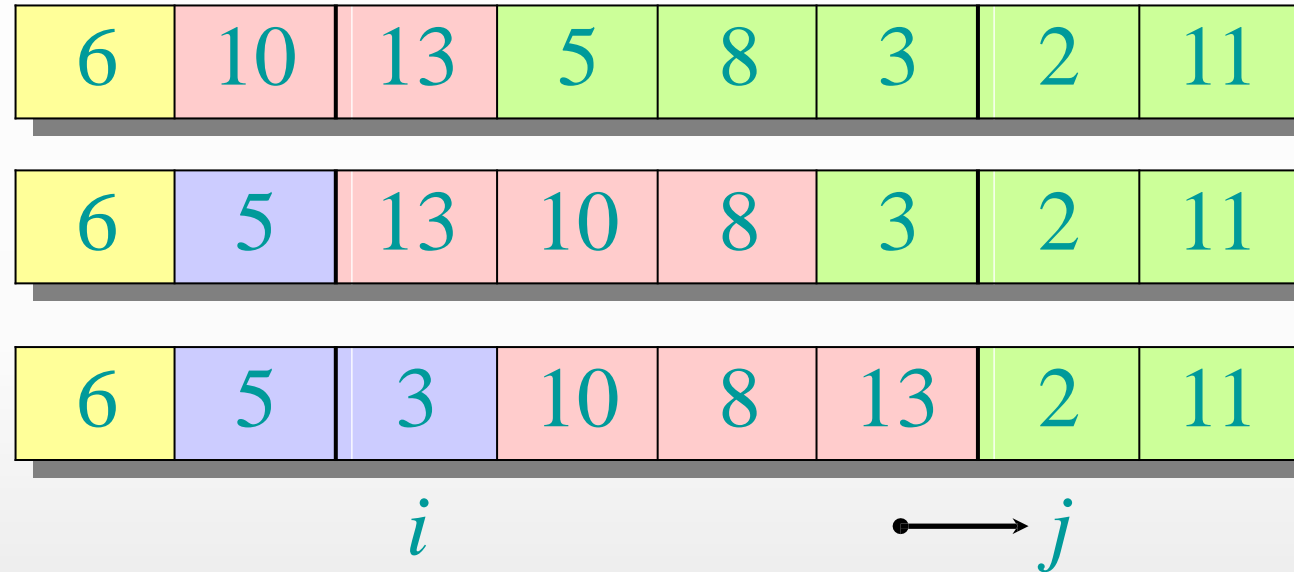


Parçalara Ayırma (Partitioning)



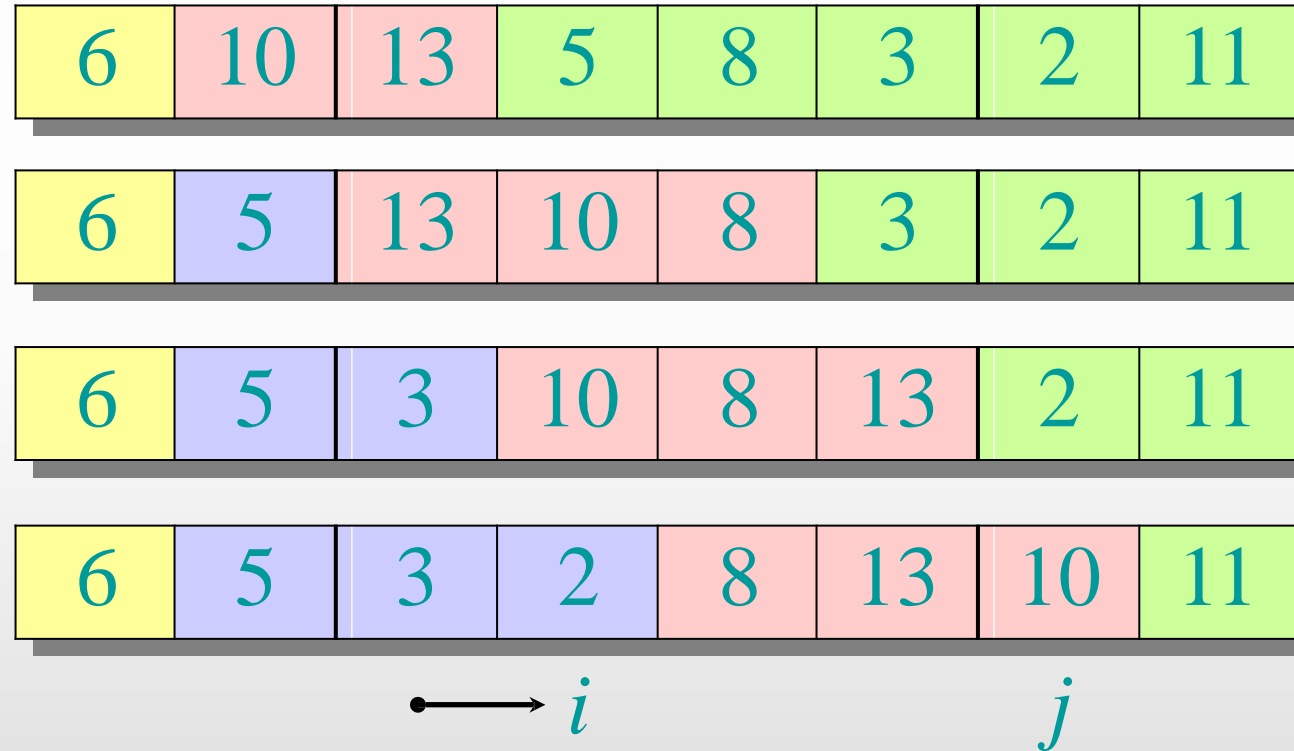


Parçalara Ayırma (Partitioning)



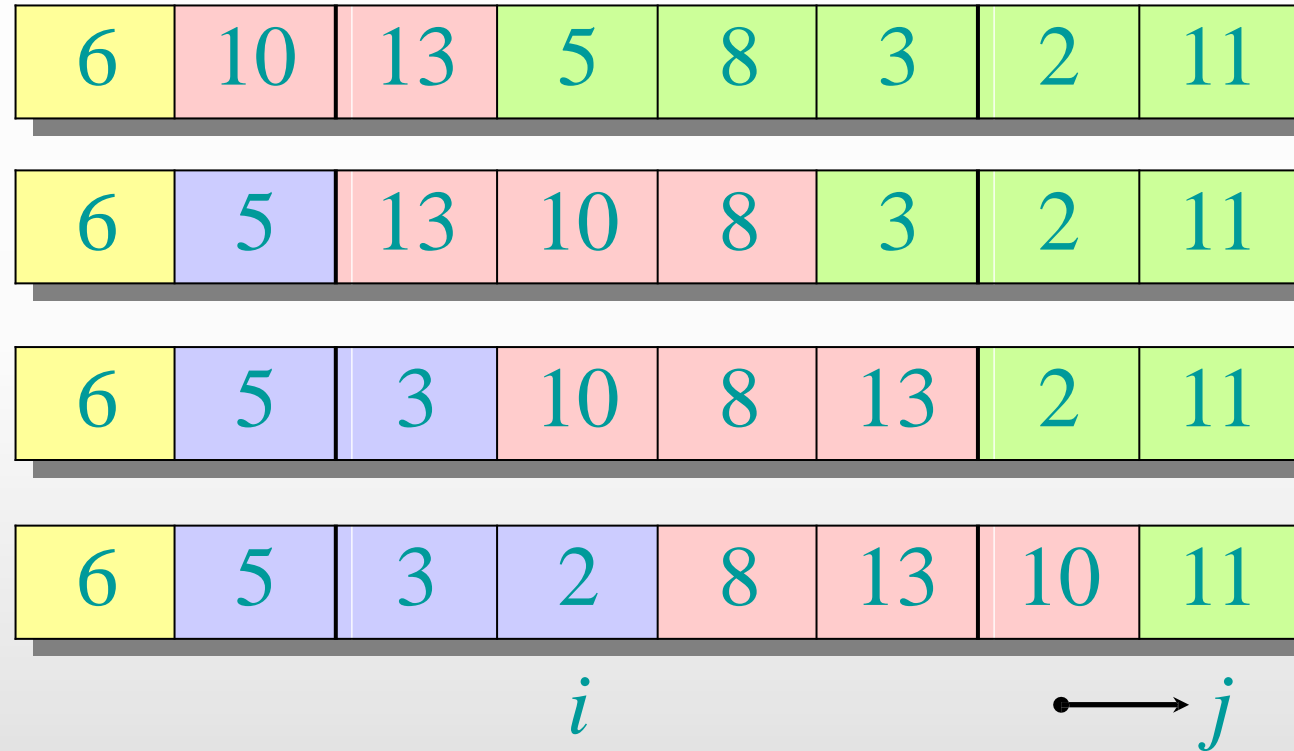


Parçalara Ayırma (Partitioning)



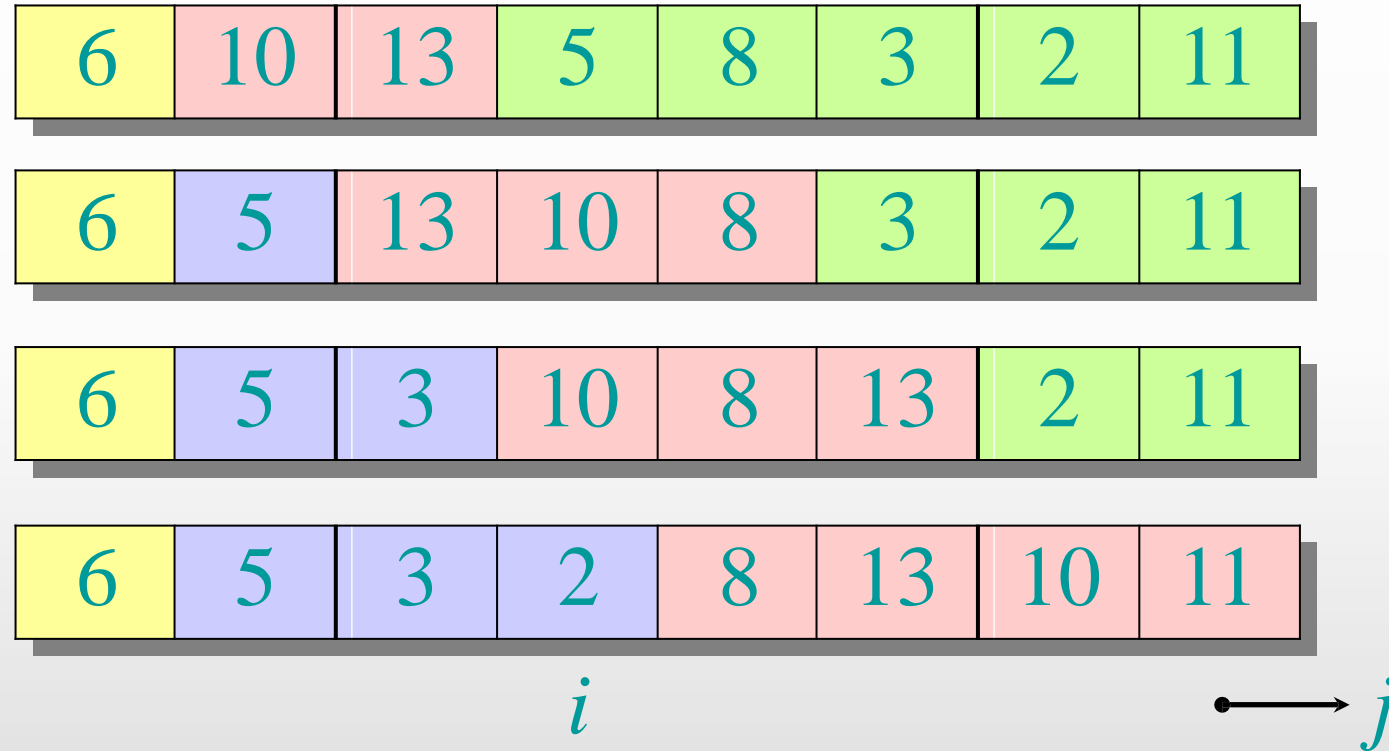


Parçalara Ayırma (Partitioning)





Parçalara Ayırma (Partitioning)





Parçalara Ayırma (Partitioning)

6	10	13	5	8	3	2	11
---	----	----	---	---	---	---	----

6	5	13	10	8	3	2	11
---	---	----	----	---	---	---	----

6	5	3	10	8	13	2	11
---	---	---	----	---	----	---	----

6	5	3	2	8	13	10	11
---	---	---	---	---	----	----	----

2	5	3	6	8	13	10	11
---	---	---	---	---	----	----	----

i



QuickSort Sözde kodu

```
QUICKSORT(A, p, r)
  if p < r
    then q = PARTITION(A, p, r)
         QUICKSORT(A, p, q-1)
         QUICKSORT(A, q+1, r)
```

İlk çağrı: QUICKSORT(A, 1, n)



Saymalı Sıralama (Counting sort)

- Elemanlar birbiriyle karşılaştırılmaz.
- **Girdi:**
 - $A[1 \dots n]$,
 - $A[j] \in \{1, 2, \dots, k\}$.
- **Çıktı:**
 - $B[1 \dots n]$, sıralı.
- **Geçici:**
 - $C[1 \dots k]$.



Saymalı Sıralama (Counting sort)

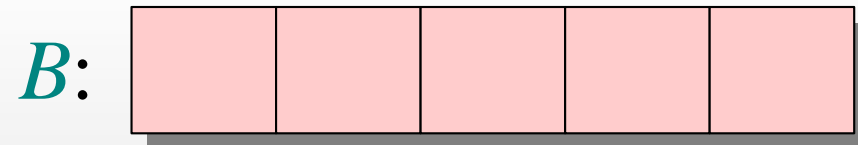
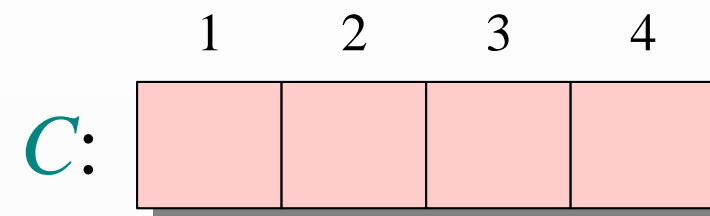
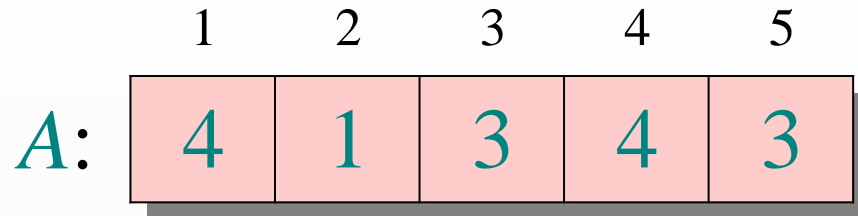
```
for i = 1 to k
  do C[i] = 0
for j = 1 to n
  do C[A[j]] = C[A[j]] + 1
for i = 2 to k
  do C[i] = C[i] + C[i-1]
for j = n downto 1
  do B[C[A[j]]] = A[j]
  C[A[j]] = C[A[j]] - 1
```

// $C[i] = |\{\text{key} = i\}|$

// $C[i] = |\{\text{key} \leq i\}|$



Saymalı Sıralama (Counting sort)



Döngü 1



	1	2	3	4	5
<i>A</i> :	4	1	3	4	3

	1	2	3	4
<i>C</i> :	0	0	0	0

<i>B</i> :					
------------	--	--	--	--	--

for $i \leftarrow 1$ **to** k
 do $C[i] \leftarrow 0$



Döngü 2

	1	2	3	4	5
A :	4	1	3	4	3

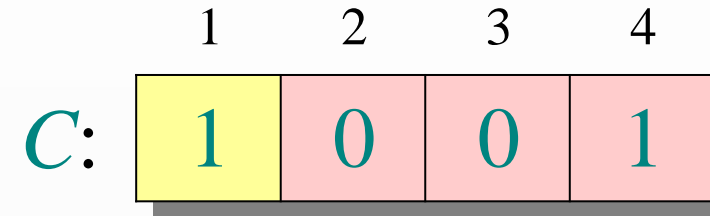
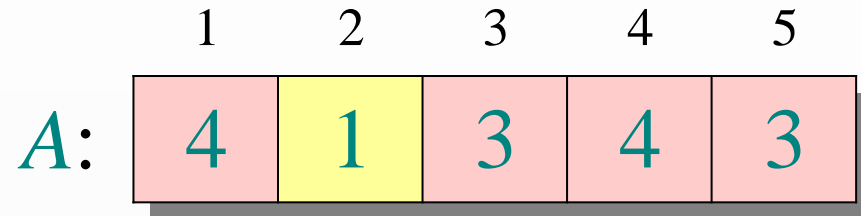
	1	2	3	4
C :	0	0	0	1

B :					
-------	--	--	--	--	--

for $j \leftarrow 1$ **to** n
 do $C[A[j]] \leftarrow C[A[j]] + 1$ $\triangleright C[i] = |\{\text{key} = i\}|$



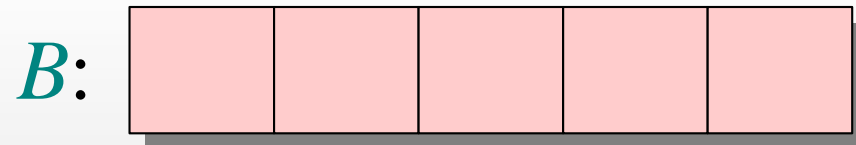
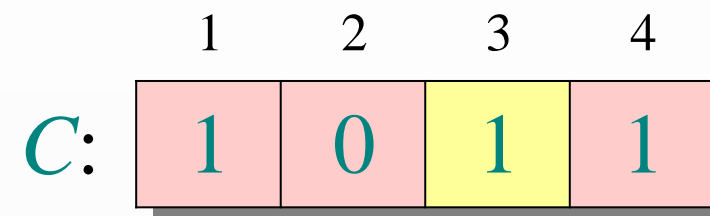
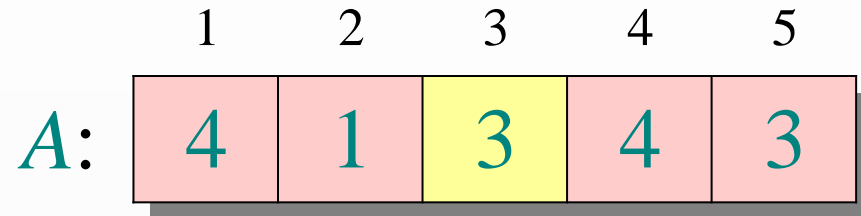
Döngü 2



for $j \leftarrow 1$ **to** n
 do $C[A[j]] \leftarrow C[A[j]] + 1 \quad \triangleright C[i] = |\{\text{key} = i\}|$



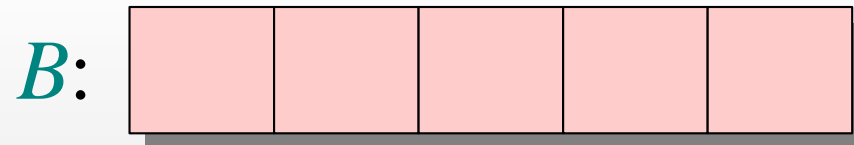
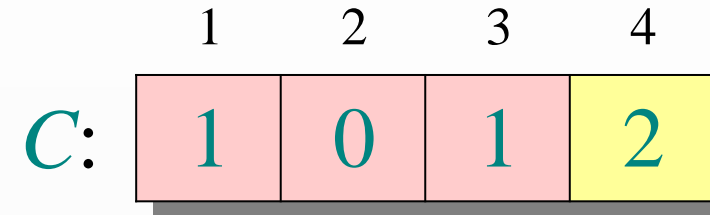
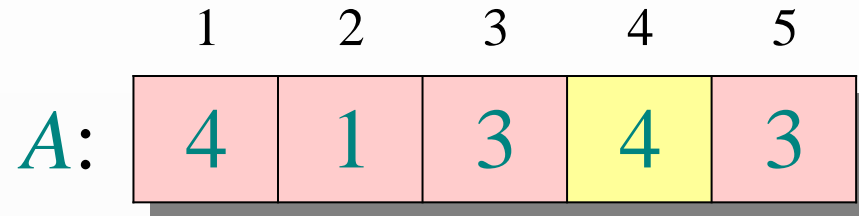
Döngü 2



for $j \leftarrow 1$ **to** n
 do $C[A[j]] \leftarrow C[A[j]] + 1 \quad \triangleright C[i] = |\{\text{key} = i\}|$



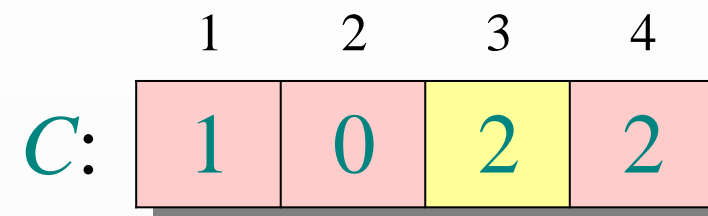
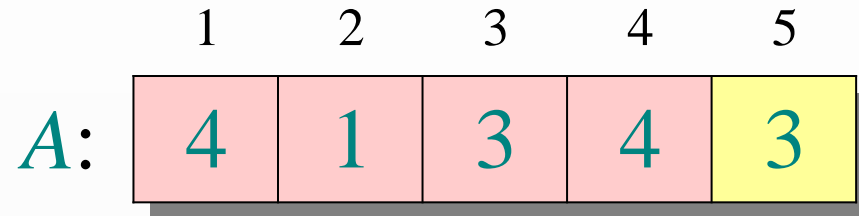
Döngü 2



for $j \leftarrow 1$ **to** n
 do $C[A[j]] \leftarrow C[A[j]] + 1 \quad \triangleright C[i] = |\{\text{key} = i\}|$



Döngü 2



for $j \leftarrow 1$ **to** n
 do $C[A[j]] \leftarrow C[A[j]] + 1$ $\triangleright C[i] = |\{\text{key} = i\}|$



Döngü 3

	1	2	3	4	5
<i>A</i> :	4	1	3	4	3

	1	2	3	4
<i>C</i> :	1	0	2	2

<i>B</i> :					
------------	--	--	--	--	--

<i>C'</i> :	1	1	2	2
-------------	---	---	---	---

for $i \leftarrow 2$ **to** k

do $C[i] \leftarrow C[i] + C[i-1]$ $\triangleright C[i] = |\{\text{key} \leq i\}|$



Döngü 3

	1	2	3	4	5
<i>A</i> :	4	1	3	4	3

	1	2	3	4
<i>C</i> :	1	0	2	2

<i>B</i> :					
------------	--	--	--	--	--

<i>C'</i> :	1	1	3	2
-------------	---	---	---	---

for $i \leftarrow 2$ **to** k

do $C[i] \leftarrow C[i] + C[i-1]$

$\triangleright C[i] = |\{\text{key} \leq i\}|$



Döngü 3

	1	2	3	4	5
<i>A</i> :	4	1	3	4	3

	1	2	3	4
<i>C</i> :	1	0	2	2

<i>B</i> :					
------------	--	--	--	--	--

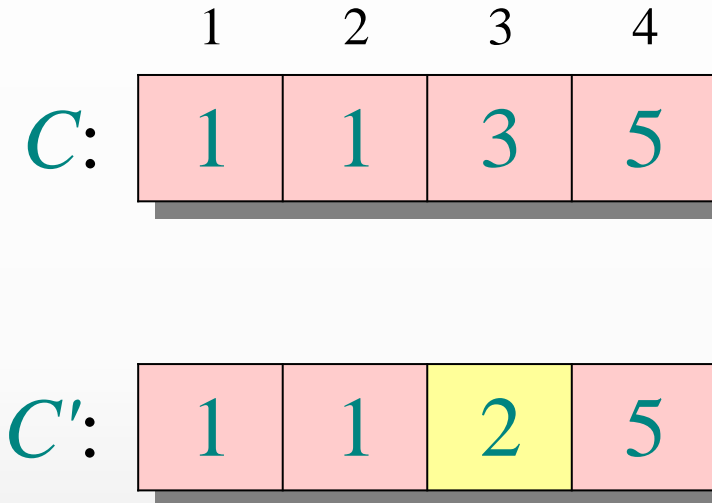
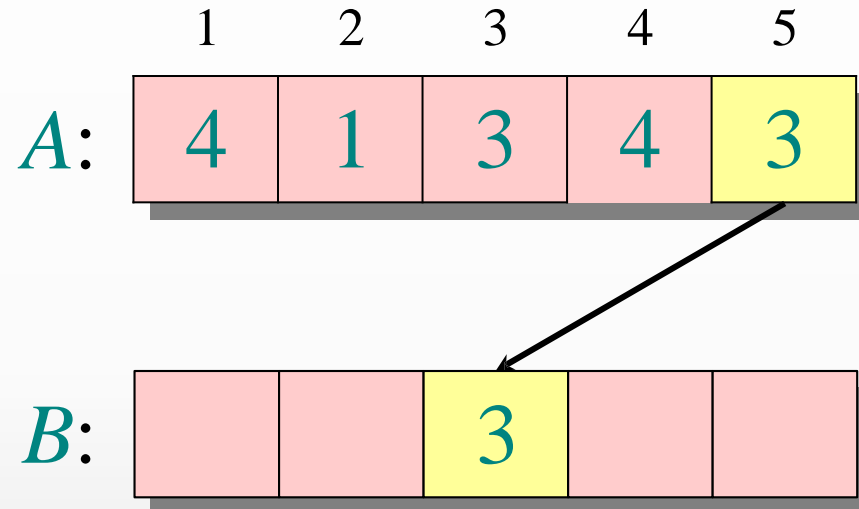
<i>C'</i> :	1	1	3	5
-------------	---	---	---	---

for $i \leftarrow 2$ **to** k

do $C[i] \leftarrow C[i] + C[i-1]$ $\triangleright C[i] = |\{\text{key} \leq i\}|$



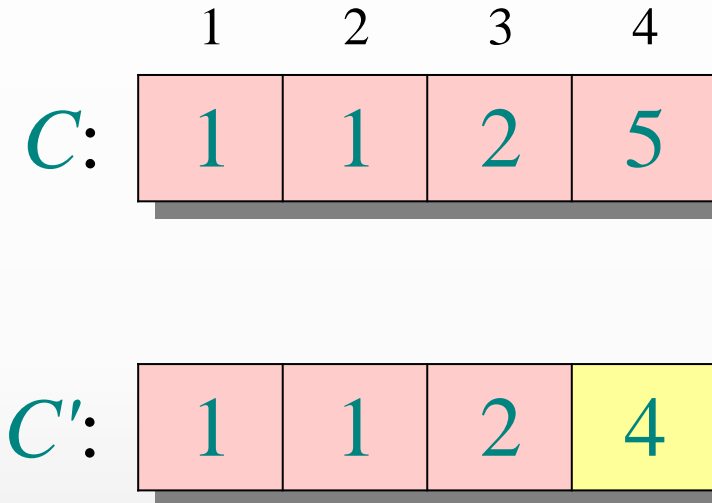
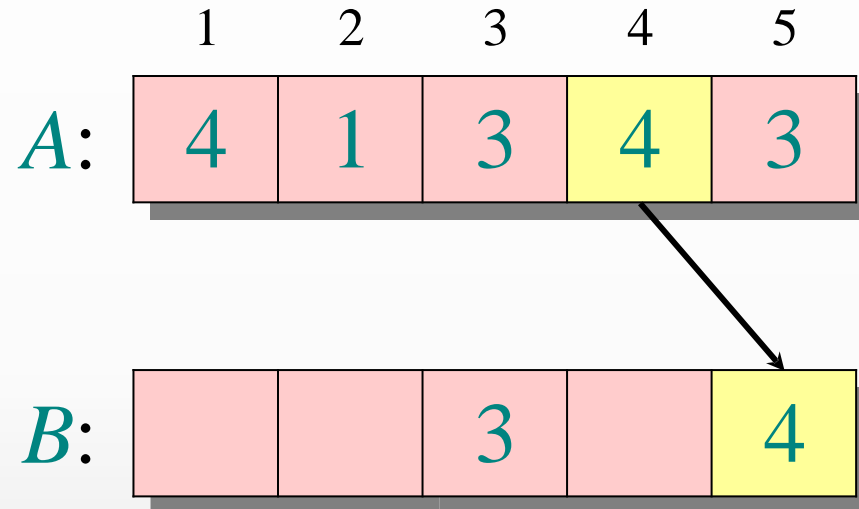
Döngü 4



```
for  $j \leftarrow n$  downto 1  
  do  $B[C[A[j]]] \leftarrow A[j]$   
      $C[A[j]] \leftarrow C[A[j]] - 1$ 
```



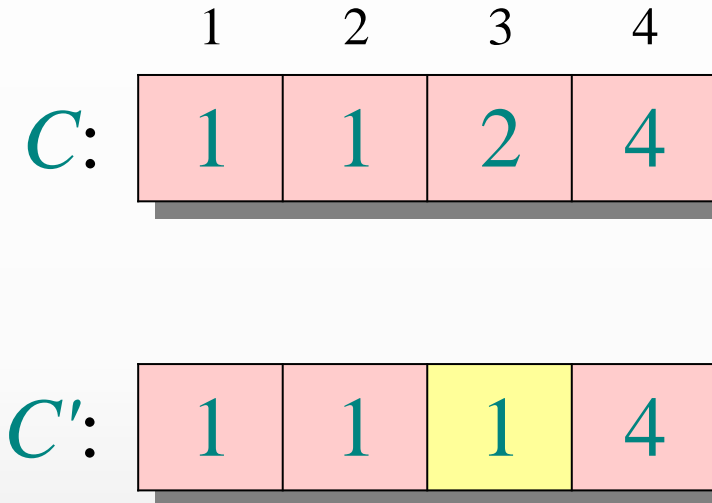
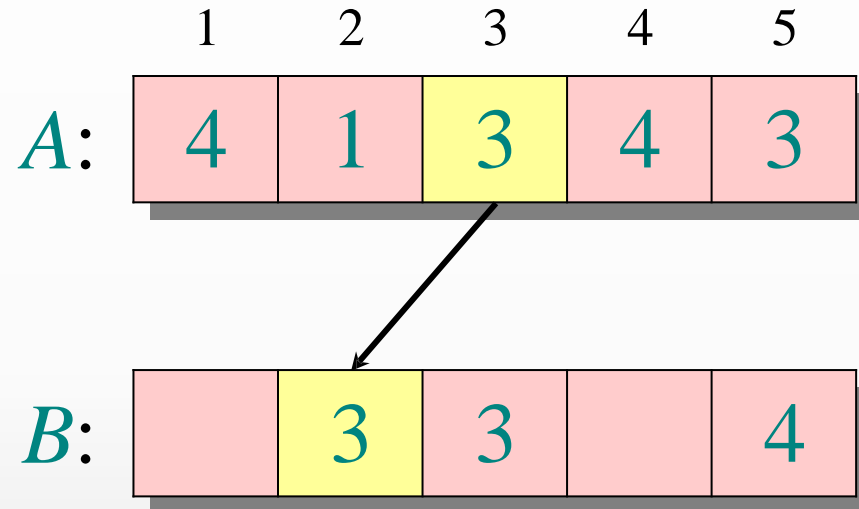
Döngü 4



```
for  $j \leftarrow n$  downto 1  
  do  $B[C[A[j]]] \leftarrow A[j]$   
      $C[A[j]] \leftarrow C[A[j]] - 1$ 
```



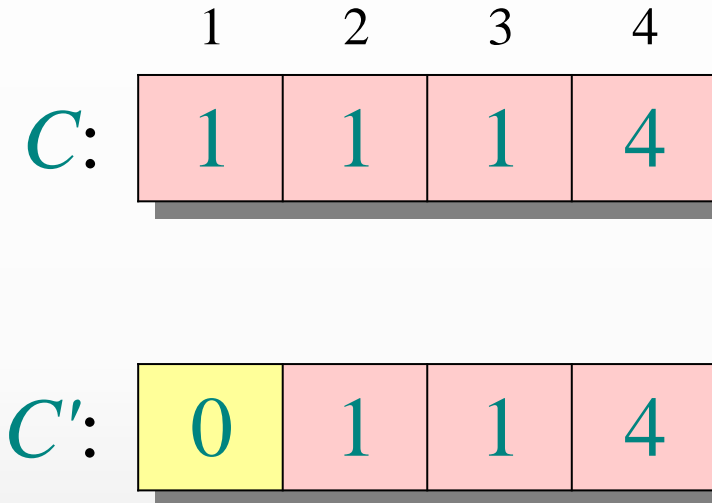
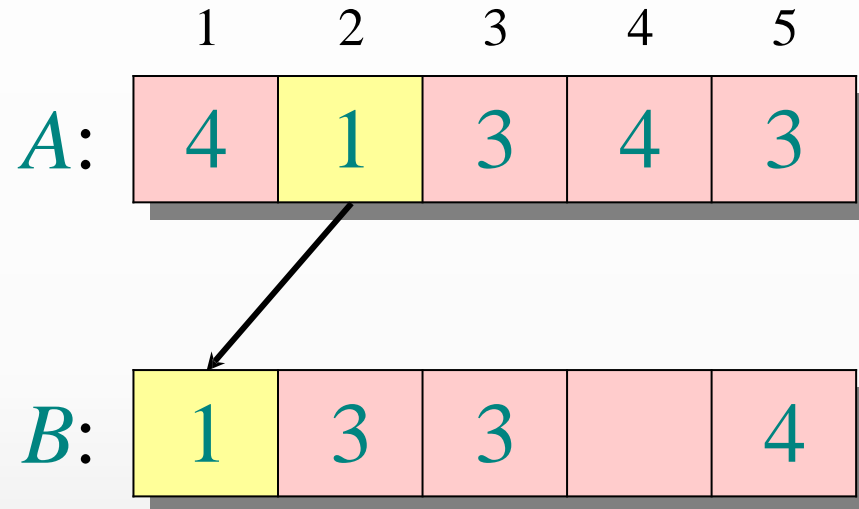
Döngü 4



```
for  $j \leftarrow n$  downto 1  
  do  $B[C[A[j]]] \leftarrow A[j]$   
      $C[A[j]] \leftarrow C[A[j]] - 1$ 
```



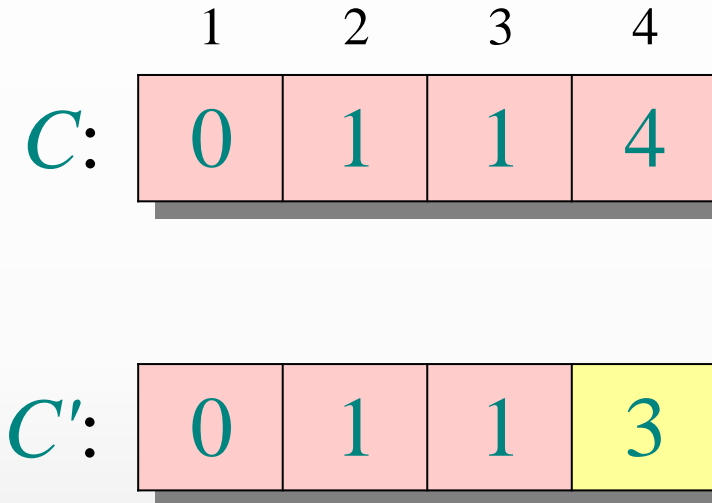
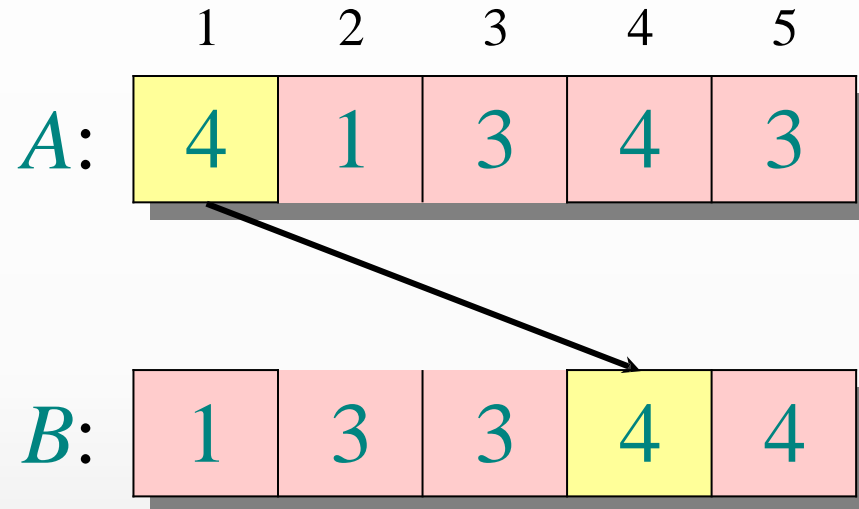
Döngü 4



```
for  $j \leftarrow n$  downto 1  
  do  $B[C[A[j]]] \leftarrow A[j]$   
      $C[A[j]] \leftarrow C[A[j]] - 1$ 
```



Döngü 4



```
for  $j \leftarrow n$  downto 1  
  do  $B[C[A[j]]] \leftarrow A[j]$   
      $C[A[j]] \leftarrow C[A[j]] - 1$ 
```



Algoritma Karmaşıklığı

$\Theta(k)$ { **for** $i \leftarrow 1$ **to** k
 do $C[i] \leftarrow 0$

$\Theta(n)$ { **for** $j \leftarrow 1$ **to** n
 do $C[A[j]] \leftarrow C[A[j]] + 1$

$\Theta(k)$ { **for** $i \leftarrow 2$ **to** k
 do $C[i] \leftarrow C[i] + C[i-1]$

$\Theta(n)$ { **for** $j \leftarrow n$ **downto** 1
 do $B[C[A[j]]] \leftarrow A[j]$
 $C[A[j]] \leftarrow C[A[j]] - 1$

$\Theta(n + k)$



SON