



동아대학교

데이터 구조

12주차: 정렬

박 경 훈

선택 정렬 (Selection Sort)

● 선택정렬

- 제자리 정렬(in-place sorting) 알고리즘의 하나
 - **입력 배열(정렬되지 않은 값) 이외에 다른 추가 메모리 요구되지 않음**
- 정렬되지 않은 데이터 중 최소값을 정렬된 배열의 제일 마지막 위치의 다음 요소와 교환하는 방식

※ 선택정렬 과정 설명

1. 주어진 배열 중에서 최소값을 찾는다.
2. 그 값을 맨 앞에 위치한 값과 교환
3. 맨 처음 위치를 뺀 나머지 리스트를 같은 방법으로 교환
- 4. 하나의 원소만 남을 때 까지 위의 1~3번 과정을 반복**

선택 정렬 (Selection Sort)

초기상태

9	6	7	3	5
---	---	---	---	---

1

9	6	7	3	5
---	---	---	---	---

↑

최솟값 탐색: 3
첫 번째 값 9와 최솟값 3을 교환

3	6	7	9	5
---	---	---	---	---

1회전 결과

2

3	6	7	9	5
---	---	---	---	---

↑

최솟값 탐색: 5
두 번째 값 6과 최솟값 5를 교환

3	5	7	9	6
---	---	---	---	---

2회전 결과

3

3	5	7	9	6
---	---	---	---	---

↑

최솟값 탐색: 6
세 번째 값 7과 최솟값 6을 교환

3	5	6	9	7
---	---	---	---	---

3회전 결과

4

3	5	6	9	7
---	---	---	---	---

↑

최솟값 탐색: 7
네 번째 값 9와 최솟값 7을 교환

3	5	6	7	9
---	---	---	---	---

4회전 결과

오름차순
완성상태

3	5	6	7	9
---	---	---	---	---

선택 정렬 (Selection Sort)

● 특징

- 장점
 - 구현이 쉽다.
- 단점
 - 데이터 개수가 많아질 수록 성능 저하

버블 정렬 (Bubble Sort)

● 버블정렬

- 서로 인접한 두 원소를 검사하여 정렬하는 알고리즘
- 선택 정렬과 기본 개념은 유사

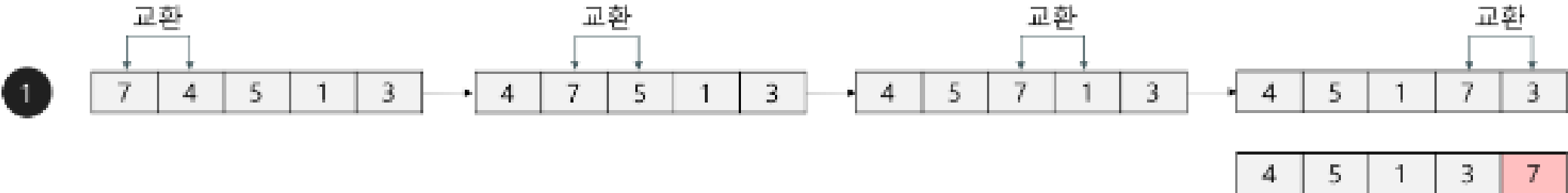
※ 버블정렬 알고리즘 과정

- 첫 번째 자료와 두 번째 자료, 두 번째 자료와 세 번째 자료와 같은 방식으로 마지막 자료까지 비교 및 교환
- 1회 순환을 마치면 가장 큰 자료가 맨 뒤로 이동
- 2회 순환 부터는 맨 끝에 있는 자료를 제외하고 수행

버블 정렬 (Bubble Sort)

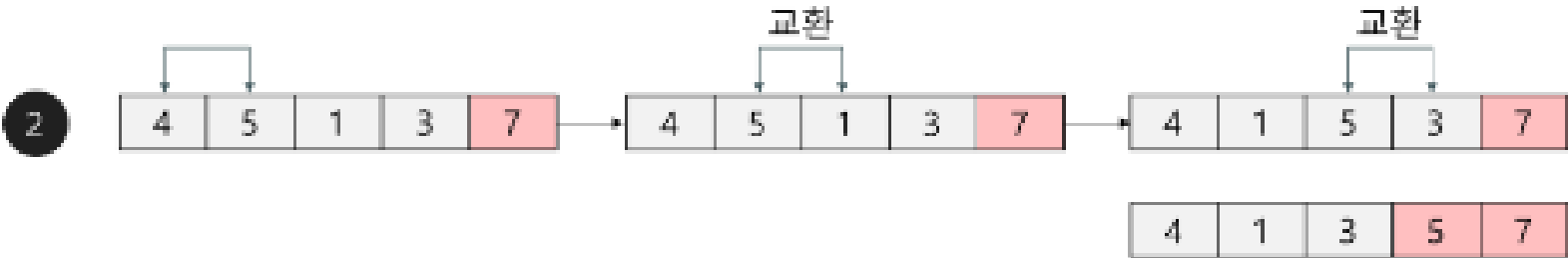
초기상태

7	4	5	1	3
---	---	---	---	---



4	5	1	3	7
---	---	---	---	---

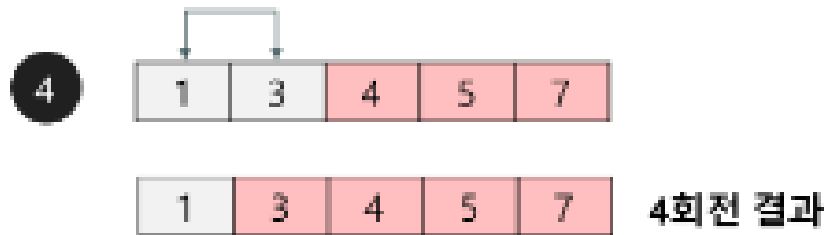
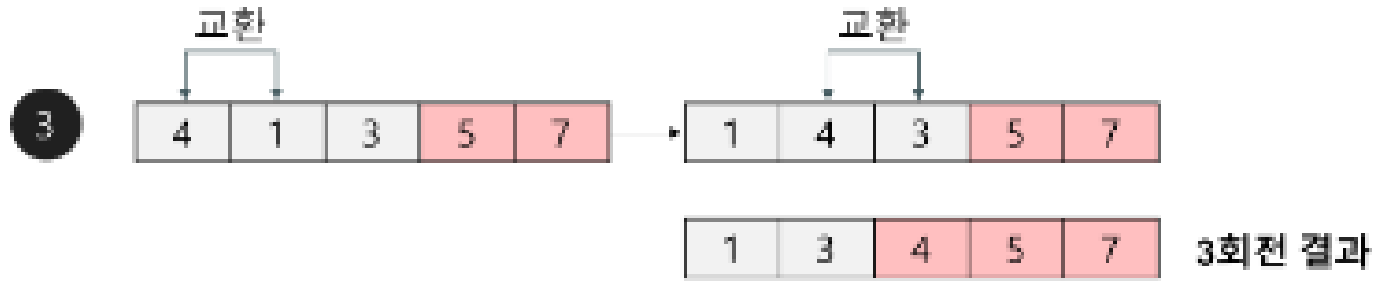
 1회전 결과



4	1	3	5	7
---	---	---	---	---

 2회전 결과

버블 정렬 (Bubble Sort)



오름차순
완성상태

1	3	4	5	7
---	---	---	---	---

삽입 정렬 (Insertion Sort)

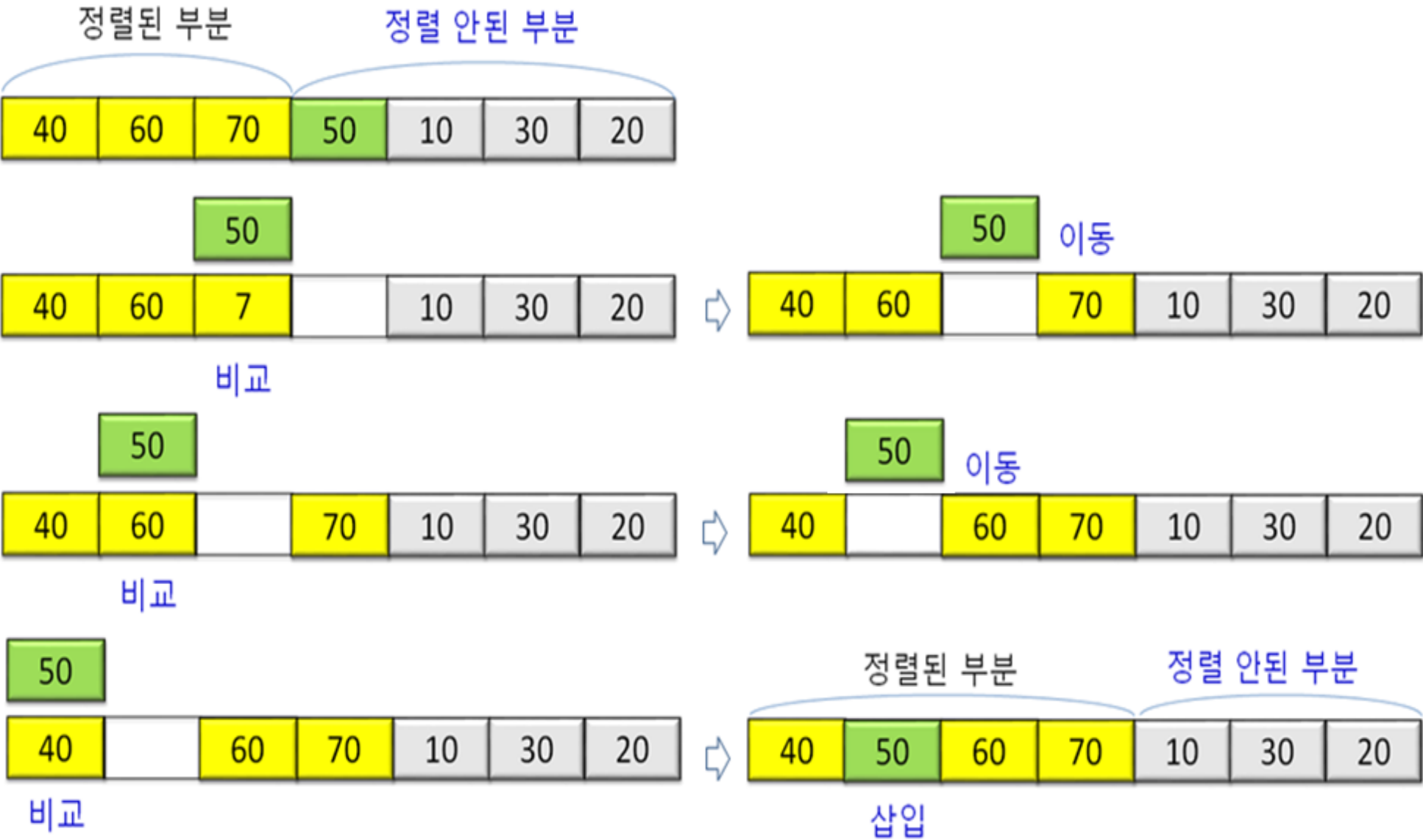
● 삽입정렬

- 자료 배열의 모든 요소를 처음부터 차례대로 이미 정렬된 배열과 비교하여, 자신의 위치를 찾아 삽입함으로써 정렬을 완성하는 알고리즘
- 매 순서 마다 해당 원소를 삽입할 수 있는 위치를 찾아 해당 위치에 넣는다.

● 특징

- 장점
 - 자료의 수가 적을 경우 알고리즘 구현이 매우 간단
 - 이미 정렬되어 있는 경우나 자료의 수가 적은 정렬에 매우 효율적
- 단점
 - 비교적 많은 레코드들의 이동을 포함
 - 자료의 수가 많고 자료의 크기가 클 경우 적합하지 않음

삽입 정렬 (Insertion Sort)



셸 정렬 (Shell Sort)

● 셸 정렬

- Donald L. Shell 이라는 사람이 제안한 방법으로 삽입정렬을 보완한 알고리즘
 - 만약, 삽입되어야 할 위치가 현재 위치에서 상당히 멀리 떨어진 곳이라면 많은 이동이 필요
- 삽입 정렬과 다르게 셸 정렬은 전체의 리스트를 한번에 정렬하지 않는다.

※ 셸 정렬 과정

1. 정렬해야 할 리스트를 일정한 기준에 따라 분류
2. 연속적이지 않은 여러 개의 부분 리스트를 생성
3. 각 부분 리스트를 삽입 정렬을 이용하여 정렬
4. 모든 부분 리스트가 정렬되면 다시 전체 리스트를 더 적은 개수의 부분 리스트로 만든 후에 알고리즘을 반복
5. 1~4까지 과정을 부분 리스트의 개수가 1이 될 때까지 반복

셸 정렬 (Shell Sort)

※ 셸 정렬 알고리즘 기본 개념

초기상태

10	8	6	20	4	3	22	1	0	15	16
----	---	---	----	---	---	----	---	---	----	----

정렬할 값의 수: 10
간격(gap) k의 초깃값: $10/2 = 5$



간격 $k=5$ 일 때의 부분 리스트를
각각 삽입 정렬로 정렬

3					10					16
	8					22				
		1					6			
			0					20		
				4					15	

1회전 결과

3	8	1	0	4	10	22	6	20	15	16
---	---	---	---	---	----	----	---	----	----	----

다음 k의 값: $(5/2)+1 = 3$

셸 정렬 (Shell Sort)

2

간격 $k=3$ 일 때의 부분 리스트들

3			0			22			15	
	8			4			6			16
		1			10			20		

간격 $k=3$ 일 때의 부분 리스트를
각각 삽입 정렬로 정렬

0			3			15			22	
	4			6			8			16
		1			10			20		

2회전 결과

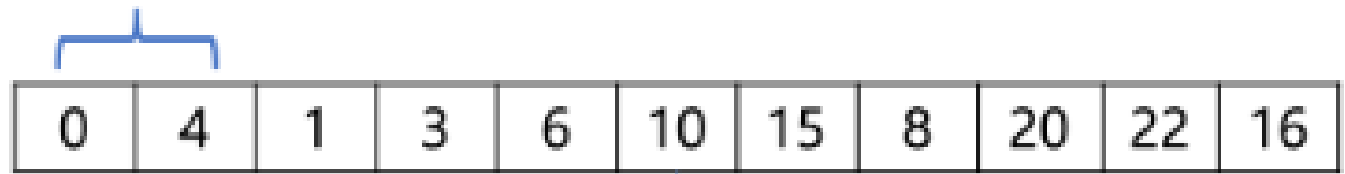
0	4	1	3	6	10	15	8	20	22	16
---	---	---	---	---	----	----	---	----	----	----

다음 k 의 값: $3/2 = 1$

셸 정렬 (Shell Sort)

3

간격 $k=1$ 일 때의 부분 리스트들



0	4	1	3	6	10	15	8	20	22	16
---	---	---	---	---	----	----	---	----	----	----

간격 $k=1$ 일 때의 부분 리스트를
각각 삽입 정렬로 정렬

0	1	3	4	6	8	10	15	16	20	22
---	---	---	---	---	---	----	----	----	----	----

3회전 결과

0	1	3	4	6	8	10	15	16	20	22
---	---	---	---	---	---	----	----	----	----	----

오름차순
완성상태

0	1	3	4	6	8	10	15	16	20	22
---	---	---	---	---	---	----	----	----	----	----

셸 정렬 (Shell Sort)

● 특징

- 장점

- 멀리 있는 원소들끼리 빠르게 비교 및 교환이 이루어진다.
- 삽입정렬, 버블정렬에 비해 정렬 속도가 빠르다.

- 단점

- 삽입정렬에 비해 구현이 어렵다.
- 부분리스트 구현을 위한 gap 설정이 잘못될 경우 비효율적 알고리즘이 될 수 있다.