

CHAPTER.23

고급 정렬 알고리즘의 원리와 구현





학습 내용

[1] 고급 정렬 알고리즘의 원리와 구현



학습 목표

- ④ 성능을 향상시킬 수 있는 정렬 방식을 설명할 수 있다.



01

고급 정렬 알고리즘의 원리와 구현

- 1) 고급 정렬
- 2) 버블 정렬(Bubble Sort)



적은 인원을 이름 순서대로 줄을 세우는 것은 어렵지 않음

하지만 여름철 해수욕장에 모인 많은 인원의 이름을
순서대로 정렬하는 것은 상당히 오랜 시간이 걸림



➔ 정렬 방식 중에서도 빠르게 정렬되는 방법을 채택해서 사용

버블 정렬의 개념

첫 번째 값부터 시작해서 바로 앞뒤 데이터를 비교하여
큰 것은 뒤로 보내는 방법을 사용하는 정렬

정렬을 위해 비교하는 모양이 거품(Bubble)처럼 생긴 것에서
이름 유래

■ 버블 정렬의 기본 개념



2] 버블 정렬

버블 정렬의 구현

4개 데이터를 버블 정렬하는 예

▪ 버블 정렬 예



버블 정렬할 데이터



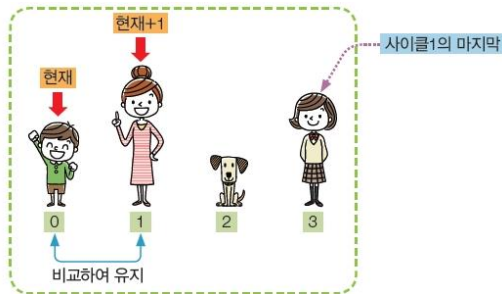
3회 사이클

버블 정렬의 구현

1 사이클1에서는 데이터 4개 중 앞부터 2개씩을 비교해서 큰 것을 뒤로 보냄

■ 버블 정렬 예 - 사이클1

반복 1회



2] 버블 정렬

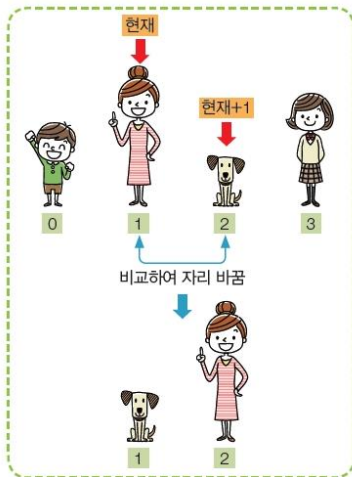


버블 정렬의 구현

1 사이클1에서는 데이터 4개 중 앞부터 2개씩을 비교해서 큰 것을 뒤로 보냄

▪ 버블 정렬 예 - 사이클1

반복 2회



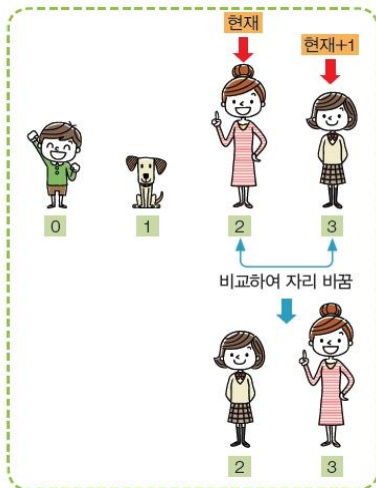
2] 버블 정렬

버블 정렬의 구현

1 사이클1에서는 데이터 4개 중 앞부터 2개씩을 비교해서 큰 것을 뒤로 보냄

▪ 버블 정렬 예 - 사이클1

반복 3회

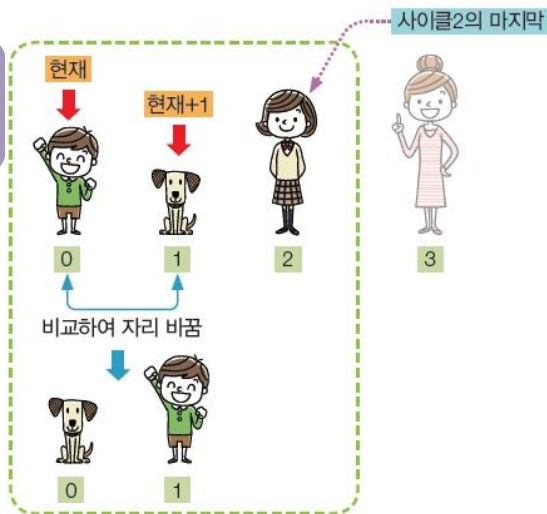


버블 정렬의 구현

2 사이클2에서는 데이터 3개 중 앞부터 2개씩을 비교해서 큰 것을 뒤로 보냄

■ 버블 정렬 예 - 사이클2

반복 1회



버블 정렬의 구현

2 사이클2에서는 데이터 3개 중 앞부터 2개씩을 비교해서 큰 것을 뒤로 보냄

■ 버블 정렬 예 - 사이클2

반복 2회



2] 버블 정렬

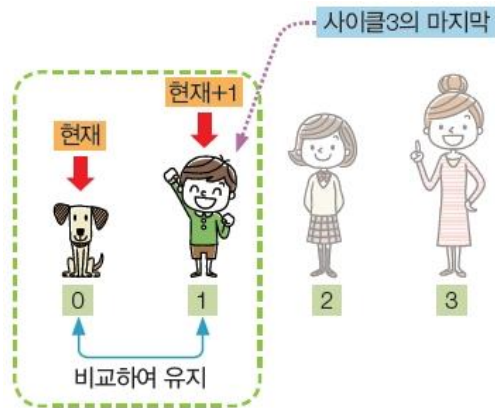


버블 정렬의 구현

3 사이클3에서는 데이터 2개 중 앞부터 2개씩을 비교해서 큰 것을 뒤로 보냄

■ 버블 정렬 예 - 사이클3

반복 1회



버블 정렬의 구현

4 모든 사이클 완료 후 정렬된 결과

- 버블 정렬 후 데이터





버블 정렬의 구현

버블 정렬 구현

```
1  ## 클래스와 함수 선언 부분 ##
2  def BubbleSort(ary) :
3      n = len(ary)
4      for end in range(n-1, 0, -1) :
5          for cur in range(0, end) :
6              if (ary[cur] > ary[cur+1]) :
7                  ary[cur], ary[cur+1] = ary[cur+1], ary[cur]
8      return ary
9
10 ## 전역 변수 선언 부분 ##
11 dataAry = [188, 162, 168, 120, 50, 150, 177, 105]
12
```

버블 정렬의 구현

버블 정렬 구현

```
13 ## 메인 코드 부분 ##  
14 print('정렬 전 -->', dataAry)  
15 dataAry = BubbleSort(dataAry)  
16 print('정렬 후 -->', dataAry)
```

실행 결과

정렬 전 --> [188, 162, 168, 120, 50, 150, 177, 105]

정렬 후 --> [50, 105, 120, 150, 162, 168, 177, 188]

버블 정렬 성능과 특이점

버블 정렬도 연산 수는 $O(n^2)$

기존에 배열이 어느 정도 정렬되어 있다면 연산 수가 급격히 줄어듦

데이터 1개를 제외하고 대부분 정렬되어 있는 배열의 경우

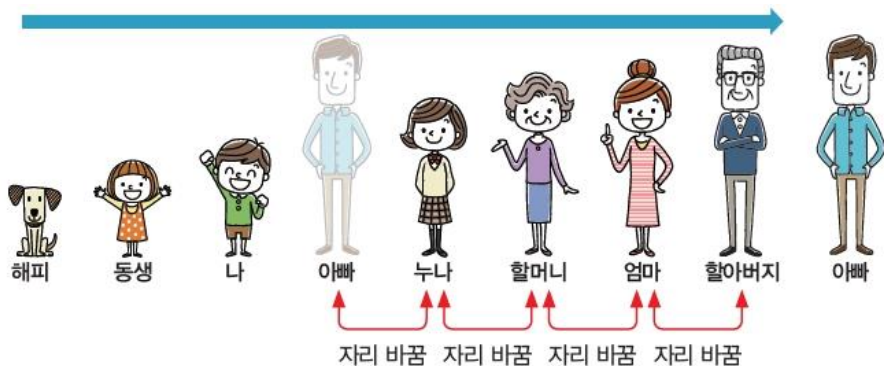
- 대부분의 데이터가 어느 정도 정렬된 초기 상태



버블 정렬 성능과 특이점

1 버블 정렬 방법에 따라 사이클1을 완료하면 정렬이 완료됨

▪ 사이클1의 작동과 결과(4회 자리 바꿈 발생)



버블 정렬 성능과 특이점

2

이미 정렬된 상태이므로 사이클2에서는 자리 바꿈이 발생되지 않음

- 사이클2의 작동과 결과(자리 바꿈 발생하지 않음)





버블 정렬 성능과 특이점

개선된 버블 정렬의 구현

```
1  ## 클래스와 함수 선언 부분 ##
2  def bubbleSort(ary) :
3      n = len(ary)
4      for end in range(n-1, 0, -1) :
5          changeYN = False
6          print('#사이클-->', ary)
7          for cur in range(0, end) :
8              if (ary[cur] > ary[cur+1]) :
9                  ary[cur], ary[cur+1] = ary[cur+1], ary[cur]
10                 changeYN = True
11             if not changeYN :
12                 break
13     return ary
14
```



버블 정렬 성능과 특이점

개선된 버블 정렬의 구현

```
15 ## 전역 변수 선언 부분 ##
16 dataAry = [50, 105, 120, 188, 150, 162, 168, 177]
17
18 ## 메인 코드 부분 ##
19 print('정렬 전 -->', dataAry)
20 dataAry = bubbleSort(dataAry)
21 print('정렬 후 -->', dataAry)
```

실행 결과

```
정렬 전 --> [50, 105, 120, 188, 150, 162, 168, 177]
#사이클--> [50, 105, 120, 188, 150, 162, 168, 177]
#사이클--> [50, 105, 120, 150, 162, 168, 177, 188]
정렬 후 --> [50, 105, 120, 150, 162, 168, 177, 188]
```

2] 버블 정렬



버블 정렬 실습

앞쪽의 소스를 수정해서 랜덤하게 0과 200 사이의 숫자 20개를 생성한 후 내림차순으로 정렬하도록 코드를 작성하자.
그리고 몇 번 만에 정렬했는지 횟수를 출력하자.

실행 결과

정렬 전 → [158, 20, 119, 39, 66, 41, 124, 151, 168, 154]
정렬 후 → [168, 158, 154, 151, 124, 119, 66, 41, 39, 20]
35회로 정렬 완료

2] 버블 정렬



버블 정렬 실습



Q1

Q2

Q1

버블 정렬에 대한 설명으로 거리가 먼 것은?

- 1 첫 번째 값부터 시작해서 바로 앞뒤 데이터를 비교하며 큰 것을 뒤로 보내는 방법을 사용한다.
- 2 키 순서, 이름 순서, 몸무게 순서 등을 정렬할 때 사용할 수 있다.
- 3 정렬은 오름차순과 내림차순으로 정렬할 수 있다.
- 4 삽입 정렬이나 선택 정렬보다 성능이 나쁘다.

Q1

Q2

Q1

버블 정렬에 대한 설명으로 거리가 먼 것은?

- 1 첫 번째 값부터 시작해서 바로 앞뒤 데이터를 비교하며 큰 것을 뒤로 보내는 방법을 사용한다.
- 2 키 순서, 이름 순서, 몸무게 순서 등을 정렬할 때 사용할 수 있다.
- 3 정렬은 오름차순과 내림차순으로 정렬할 수 있다.
- ✓ 4 삽입 정렬이나 선택 정렬보다 성능이 나쁘다.

정답

4 삽입 정렬이나 선택 정렬보다 성능이 나쁘다.

해설

버블 정렬도 삽입 정렬이나 선택 정렬과 마찬가지로 연산 수는 $O(n^2)$ 입니다.

Q1

Q2

Q2

버블 정렬이 선택 정렬이나 삽입 정렬보다 성능이 더 좋은 경우는?


- 1 데이터가 이미 정렬이 거의 되어 있고, 중간에 정렬되지 않은 데이터가 하나 있을 때
- 2 데이터가 정렬이 전혀 안 되어 있는 경우
- 3 오름차순 정렬할 때
- 4 내림차순 정렬할 때

Q1

Q2

Q2

버블 정렬이 선택 정렬이나 삽입 정렬보다 성능이 더 좋은 경우는?

- 1  데이터가 이미 정렬이 거의 되어 있고, 중간에 정렬되지 않은 데이터가 하나 있을 때
- 2 데이터가 정렬이 전혀 안 되어 있는 경우
- 3 오름차순 정렬할 때
- 4 내림차순 정렬할 때

정답

- 1 데이터가 이미 정렬이 거의 되어 있고, 중간에 정렬되지 않은 데이터가 하나 있을 때

해설

버블 정렬은 기존 배열이 어느 정도 정렬이 되어 있다면 연산 수가 급격히 줄어듭니다.

고급 정렬 알고리즘의 원리와 구현

☑ 버블 정렬의 개념

- 첫 번째 값부터 시작해서 바로 앞뒤 데이터를 비교하여 큰 것은 뒤로 보내는 방법을 사용하는 정렬
- 정렬을 위해 비교하는 모양이 거품 (Bubble) 처럼 생긴 것에서 이름 유래



고급 정렬 알고리즘의 원리와 구현

㉞ 버블 정렬 성능과 특이점

- 버블 정렬도 연산 수는 $O(n^2)$
- 기존에 배열이 어느 정도 정렬되어 있다면 연산 수가 급격히 줄어듦