컴퓨팅사고력 향상을 위한

문제해결과 알고리즘

1교시 : 자료정렬알고리즘

양 숙 희

INTRO

정렬은 알고리즘의 시작으로 정렬은 컴퓨터가 발달하면서 가장 많이 연구되고 개발되어온 알고리즘 분야이다.

데이터를 컴퓨터에 저장하는 이유는 저장 후 사용을 하기 위해서이다. 즉 필요한 데이터를 검색하기 위해서이다. 그런데 이 리스트의 데이터를 아무렇게나 입력 된 순서대로 둔다면 검색 시간이 어떻게 될까? 컴퓨터에서 다루는 데이터의 수는 100, 1000개일 수도 있지만 1백 만개, 1억 개 인 경우가 훨씬 더 많다. 이 데이터를 이름순서나, 주민등록번호순으로 정리해 둔 다면 찾는 과정이 훨씬 쉬워진다.

정렬은 데이터 검색을 빠르게 해준다.

학습목표

1. 정렬 알고리즘의 개념을 이해 할 수 있다.

2. 정렬 알고리즘의 종류별 특징을 이해 할 수 있다.

3. 정렬 알고리즘을 선택하여 문제를 해결 할 수 있다

자료 정렬이란

- **자료 정렬이란**
  - 자료를 원하는 기준에 맞게 순서를 재배치
  - 자료 처리의 효율성을 향상시키기 위한 작업
- **자료 정렬의 장점 => 자료 정렬을 자료 처리의 기본에 해당**
  - 많은 양의 자료 속에서 원하는 정보를 찾고자 할 때 자료가 정렬되어 있다면 훨씬 빠르게 처리할 수 있음
  - 자료의 양이 방대할 때 원하는 자료를 찾아서 처리해야 한다면, 자료 정렬은 자료 검색을 위한 우선 단계

자료 정렬을 위한 정렬 기준

- **순서 없이 나열된 자료를 정렬하기 위해서는 정렬 기준을 정해야 함**
  - 오름차순(ascending order)
  - 내림차순(descending order)

- **자료를 정렬하는 데 기준이 되는 값을 키(key) 값이라 함**

- **자료 정렬을 효율적으로 실행하기 위해서는 정렬 기준이 될 키(key) 값의 선택과 정렬 순서 방식에 대한 지정이 필요함**

자료 정렬을 위한 정렬 기준의 예

예: 전화번호부

- 전화번호로 정렬을 한다면?
- 사람 이름 또는 기관명이 키(key) 값에 해당
- 가나다순으로 지정되었다면, 오름차순이 적용된 정렬 방식에 해당

자료 정렬 과정

자료 정렬의 과정은 3단계로 구성

- 1단계 정렬 기준 정하기
  - ✓ 주어진 자료 정렬 기준을 설정
- 2단계 자료 비교하기
  - ✓ 기준에 맞추어 자료를 비교
- 3단계 자료 이동하기
  - ✓ 비교 결과에 따라 해당하는 위치로 이동

자료 정렬 과정의 예

예: 키 순서로 정렬하기

- 1단계 정렬 기준 정하기 : 키 작은 학생
- 2단계 자료 비교하기 : 학생들의 키를 각각 비교
- 3단계 자료 이동하기 : 올바른 위치로 이동시켜 정렬 완성

실생활에서 자료 정렬 활용

실생활에서 접할 수 있는 정렬된 자료 : 사전

- 사전의 내용이 정렬되어 있지 않다면 원하는 단어를 찾기란 불가능
- 사전에서의 키(key) 값 : 해당 단어

자료 정렬 알고리즘 : 선택 정렬

선택 정렬(Selection Sort)

- 오름차순의 선택 정렬은 배열된 자료의 값을 모두 비교한 후
- 가장 작은 값을 선택하여 배열의 첫 번째 자리의 값과 위치를 교환
- 가장 작은 값부터 정렬
- 두 번째 값부터 마지막 n번째 값을 대상으로 같은 작업을 반복하여 실행
- 전체 자료를 정렬하기 위해서는 자료의 개수가 n이라면
    - n-1번의 반복을 실행한 후 정렬이 완료

자료 정렬 알고리즘 : 선택 정렬의 예

초기 데이터 : 8 6 3 2 4

0단계	8	6	3	2	4
1단계	2	8	6	3	4
2단계	2	3	8	6	4
3단계	2	3	4	8	6
4단계	2	3	4	6	8

자료 정렬 알고리즘 : 삽입 정렬

삽입 정렬(Insertion Sort)

- 자료 배열의 모든 요소를 앞에서부터 차례대로 정렬하는 방식
- 이미 정렬된 배열 부분의 내용과 자신의 값을 비교하며 자신의 위치를 찾아 삽입
- 자료를 하나씩 올바른 위치에 삽입하여 정렬하는 방식이므로
  - 정렬하는 과정 중에도 정렬하는 순번의 배열까지는 정렬 상태를 유지

자료 정렬 알고리즘 : 삽입 정렬의 예

초기 데이터 : 8 6 3 2 4

0단계	8	6	3	2	4
1단계	6	8	3	2	4
2단계	3	6	8	2	4
3단계					
4단계	2	3	4	6	8

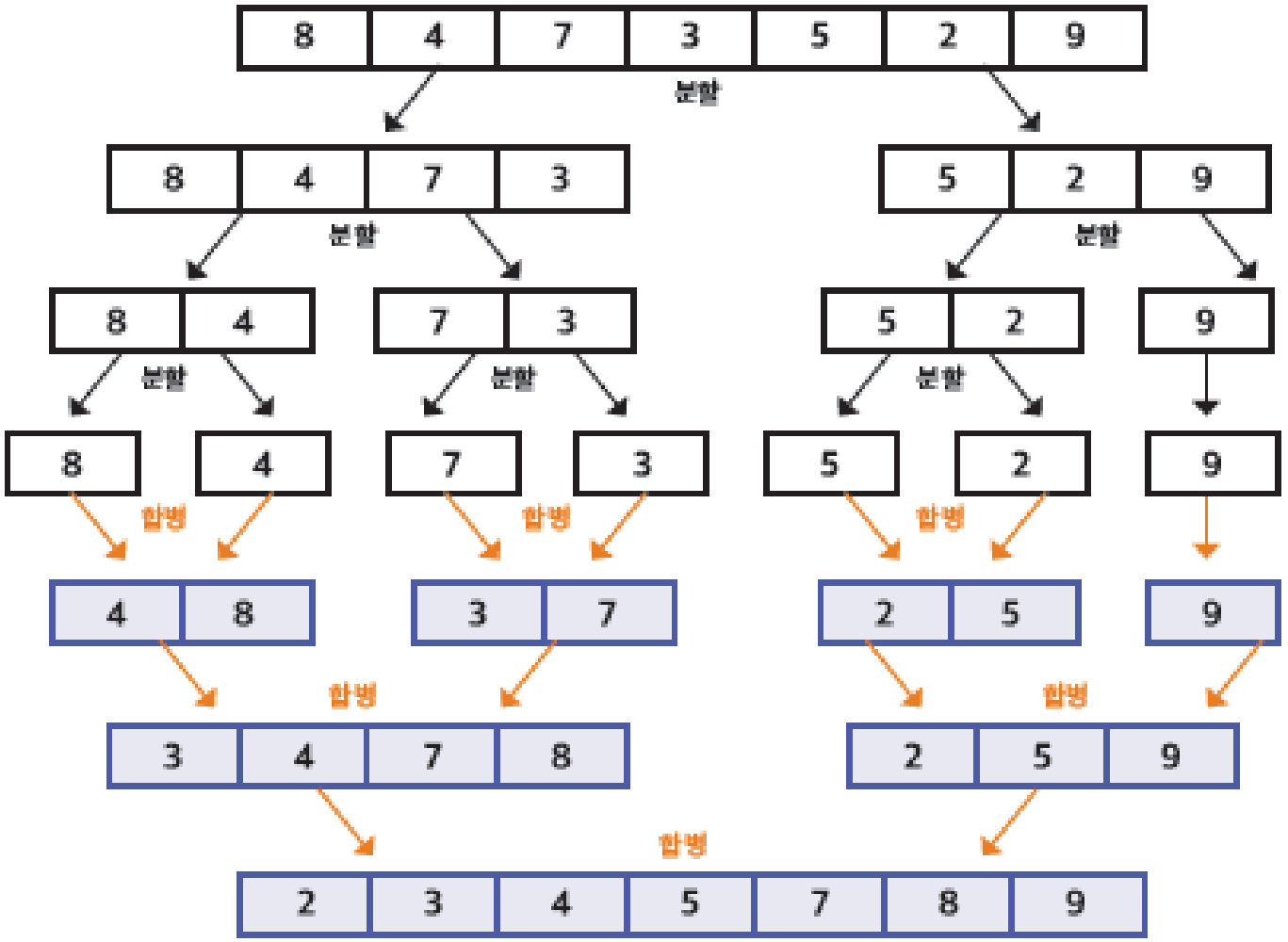
자료 정렬 알고리즘 : 합병(병합) 정렬

합병 정렬(Merge Sort)

- 자료를 두 개의 균등한 크기로 분할하는 작업부터 시작
- 분할 작업을 최소 단위까지 반복한 후 최소 단위의 두 자료를 정렬 기준에 맞게 합병
- 전체 자료가 정렬될 때까지 반복하여 병합하는 방식의 알고리즘
- 전체의 문제를 한 번에 푸는 방식이 아니라 문제를 작은 두 개의 문제로 분리하고, 다시 그 문제를 반복 분리하여 최소 단위로 분리시킨 뒤 문제를 해결하고, 다시 그 해결을 반복 적용하여 전체를 해결하는 방식

자료 정렬 알고리즘 : 합병 정렬의 예

정렬해야 할 자료: 8 4 7 3 5 2 9



합병 1단계

합병 2단계

합병 3단계

자료 정렬 알고리즘 : 버블 정렬

버블 정렬(Bubble Sort)

- 앞에서부터 하나씩 값을 비교하여 뒤로 이동하는 정렬 방식
- 가장 큰 값을 맨 끝에 위치시키기 위하여 두 개의 인접 값을 반복하여 비교한 후 큰 값이 뒤에 오도록 반복 교체하는 작업을 수행

자료 정렬 알고리즘 : 버블 정렬의 예

초기 데이터 : 8 6 3 2 4

0단계	8	6	3	2	4
1단계	6	3	2	4	8
2단계	3	2	4	6	8
3단계	2	3	4	6	8
4단계	2	3	4	6	8

자료 정렬 알고리즘의 효율성

- **여러 가지 자료 정렬 알고리즘 가운데 주어진 상황에 가장 적합한 알고리즘을 선택할 시 고려 사항**

  ① 자료의 양

  ② 사용 가능한 기억 공간의 크기

  ③ 정렬을 위한 자료 이동 빈도 수

- **정렬을 위한 자료 이동 빈도 수가 적은 효율성이 높은 정렬 알고리즘을 선택**

SWAP 알고리즘

```
# swap algorithm


a=int(input('num1:'))
b=int(input('num2:'))
print('swap before')
print(a,b)
# swap
tmp=a
a=b
b=tmp

print('swap after')
print(a,b)
```

```
a,b=b,a
```

리스트 교환

```
>>> a=[10,20]
>>> b=[20,10]
>>> a,b=b,a
>>> a
[20, 10]
>>> b
[10, 20]
```

선택정렬 알고리즘

```
선택정렬 전
[8, 6, 3, 2, 4]

선택정렬 후
1 단계: [2, 8, 6, 3, 4]
------------------------------
2 단계: [2, 3, 8, 6, 4]
------------------------------
3 단계: [2, 3, 4, 8, 6]
------------------------------
4 단계: [2, 3, 4, 6, 8]
------------------------------
총 교환 횟수 8
```

```
# 함수정의

def selection_sort(a):                  #
    n=len(a)
    cnt=0
    for i in range(      ):             #
        for j in range(          ):     #
            if                  :       #
                a[i],a[j]=a[j],a[i]
                cnt=cnt+1               #
        print(i+1,'단계:',a)            #
        print('-'*30)
    print('총 교환 횟수', cnt)
```

버블정렬 알고리즘

```
버블정렬 전
[8, 6, 3, 2, 4]

버블정렬 후
1 단계: [6, 3, 2, 4, 8]
------------------------------
2 단계: [3, 2, 4, 6, 8]
------------------------------
3 단계: [2, 3, 4, 6, 8]
------------------------------
4 단계: [2, 3, 4, 6, 8]
------------------------------
총 교환 횟수 8
```

```
# 버블정렬

def bubble_sort(a):
    n=len(a)
    cnt=0
    for i in range(      ):                    # (리스트길이-1)만큼 반복
        for j in range(r      ):
            if                  :               # 앞의 자료가 뒤의 자료보
                a[j],a[j+1]=a[j+1],a[j] # 서로 교환
                cnt+=1
        print(i+1,'단계:',a)                    # 단계별 출력
        print('-'*30)
    print('총 교환 횟수', cnt)


# 메인코드
a=[8,6,3,2,4]          # 자료 리스트 선언
print('버블정렬 전')
print(a)
print()
print('버블정렬 후')
bubble_sort(a)    # 버블정렬 함수 호출
```

자료 정렬 알고리즘

- 자료 정렬이란
- 자료 정렬을 위한 정렬 기준
- 자료 정렬 과정
- 실생활에서 자료 정렬 활용
- 자료 정렬 알고리즘의 종류
  - 선택 정렬
  - 삽입 정렬
  - 합병 정렬
  - 버블 정렬
- 자료 정렬 알고리즘의 효율성

수고하셨습니다.