

1 | 검색의 개념과 검색 방법

1 검색(Search)

- ▶ 컴퓨터에 저장한 자료 중에서 원하는 항목을 찾는 작업
- ▶ 필통에서 원하는 필기구를 찾는 것, 영어 사전에서 단어를 찾는 것, 사람들이 많이 모인 청계천에서 친구를 찾는 것, 인터넷에서 사용중인 영화를 찾는 것 모두 검색임

1 검색(Search)

- ▶ 자료를 만들고 저장하고 정렬하는 이유는 자료를 사용하기 위해서이고, 사용하려면 자료 중에서도 원하는 항목을 찾아야 하는데 이것이 바로 검색
 - 검색 성공 - 원하는 항목을 찾은 경우
 - 검색 실패 - 원하는 항목을 찾지 못한 경우
- ▶ 탐색 키를 가진 항목을 찾는 것
 - 탐색 키(Search key)
: 자료를 구별하여 인식할 수 있는 키
- ▶ 삽입/삭제 작업에서의 검색
 - 원소를 삽입하거나 삭제할 위치를 찾기 위해서 검색 연산 수행

2 검색 방법

- ▶ 수행 위치에 따른 분류
 - 내부 검색(Internal Search)
: 메모리 내의 자료에 대해서 검색 수행
 - 외부 검색(External Search)
: 보조 기억 장치에 있는 자료에 대해서 검색 수행
- ▶ 검색 방식에 따른 분류
 - 비교 검색 방식(Comparison search method)
 - 계산 검색 방식(Non-comparison method)

2 검색 방법

[검색 방법에 따른 분류]

검색 방법	설명	종류
비교 검색	검색 대상의 키를 비교하여 검색한다.	순차 검색, 이진 검색, 트리 검색
계산 검색	계수적인 성질을 이용한 계산으로 검색한다.	해싱

※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

2 | 순차 검색 방법

1 검색 방법

- ▶ 순차 검색(Sequential search), 선형 검색(Linear search)
 - 일렬로 된 자료를 처음부터 마지막까지 순서대로 검색하는 방법
 - 가장 간단하고 직접적인 검색 방법
 - 배열이나 연결 리스트로 구현된 순차 자료 구조에서 원하는 항목을 찾는 방법
 - 검색 대상 자료가 많은 경우에 비효율적이지만 알고리즘이 단순하여 구현이 용이함
 - 검색해야 할 자료가 정렬된 상태인지 아닌지에 따라 검색 실패를 판단하는 시점이 달라짐

2 정렬되어 있지 않은 자료를 순차 검색

▶ 검색 방법

- 첫 번째 원소부터 시작하여 마지막 원소까지 순서대로 키 값이 일치하는 원소가 있는지를 비교하여 찾음
 - 키 값이 일치하는 원소를 찾으면 그 원소가 몇 번째 원소인지를 반환
 - 마지막 원소까지 비교하여 키 값이 일치하는 원소가 없으면 찾은 원소가 없는 것으로 검색 실패

2 | 순차 검색 방법

2 정렬되어 있지 않은 자료를 순차 검색

[정렬되어 있지 않은 자료에서의 순차 검색 예]

8	30	1	9	11	19	2
---	----	---	---	----	----	---

(a) 정렬되어 있지 않은 자료의 예

① 8≠9	8	30	1	9	11	19	2
-------	---	----	---	---	----	----	---

② 30≠9	8	30	1	9	11	19	2
--------	---	----	---	---	----	----	---

③ 1≠9	8	30	1	9	11	19	2
-------	---	----	---	---	----	----	---

④ 9=9	8	30	1	9	11	19	2
-------	---	----	---	---	----	----	---

(b) 검색 성공의 예 : 9 검색

① 8≠6	8	30	1	9	11	19	2
-------	---	----	---	---	----	----	---

② 30≠6	8	30	1	9	11	19	2
--------	---	----	---	---	----	----	---

③ 1≠6	8	30	1	9	11	19	2
-------	---	----	---	---	----	----	---

④ 9≠6	8	30	1	9	11	19	2
-------	---	----	---	---	----	----	---

⑤ 11≠6	8	30	1	9	11	19	2
--------	---	----	---	---	----	----	---

⑥ 19≠6	8	30	1	9	11	19	2
--------	---	----	---	---	----	----	---

⑦ 2≠6	8	30	1	9	11	19	2
-------	---	----	---	---	----	----	---

(c) 검색 실패의 예 : 6 검색

※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

2 정렬되어 있지 않은 자료를 순차 검색

- ▶ 정렬되지 않은 자료의 순차검색 알고리즘
 - 첫 번째 원소부터 시작하여 마지막 원소까지 순서대로 키 값이 일치하는 원소가 있는지를 비교하여 찾음

알고리즘 10-1 정렬되지 않은 자료의 순차 검색

```
sequentialSearch1(a[], n, key)
    i ← 0;
    while (i < n and a[i] ≠ key) do {
        i ← i + 1;
    }
    if (i < n) then return i;
    else return -1;
end sequentialSearch1()
```

※출처: 이지영(2016).
IT CookBook, C로 배우는 쉬운
자료구조(개정3판). 한빛미디어

2 정렬되어 있지 않은 자료를 순차 검색

▶ 정렬되지 않은 자료의 순차검색 알고리즘

- 비교횟수 - 찾고자 하는 원소의 위치에 따라 결정
 - 찾는 원소가 첫 번째 원소라면 비교횟수는 1번,
두 번째 원소라면 비교횟수는 2번,
세 번째 원소라면 비교횟수는 3번,
찾는 원소가 i번째 원소라면 i번
 - 평균 비교 횟수 : $\frac{1}{n} \sum_{i=1}^n i = \frac{1}{n} \times \frac{n(n+1)}{2} = \frac{n+1}{2}$
 - 평균 시간 복잡도 : O(n)

3 정렬된 자료를 순차 검색

- ▶ 원소의 키 값이 찾는 키 값보다 크면 찾는 원소가 없는 것이므로 더 이상 검색을 수행하지 않아도 검색 실패를 알 수 있음

[정렬된 자료에서의 순차 검색 예]

1	2	8	9	11	19	29
---	---	---	---	----	----	----

(a) 정렬된 자료의 예

① 1 < 9	1	2	8	9	11	19	29
② 2 < 9	1	2	8	9	11	19	29
③ 8 < 9	1	2	8	9	11	19	29
④ 9 = 9	1	2	8	9	11	19	29

(b) 검색 성공의 예 : 9 검색

① 1 < 6	1	2	8	9	11	19	29
② 2 < 6	1	2	8	9	11	19	29
③ 8 > 6	1	2	8	9	11	19	29

→ 검색 종료

(c) 검색 실패의 예 : 6 검색

※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

3 정렬된 자료를 순차 검색

▶ 정렬된 자료의 순차검색

- 비교횟수 - 찾고자 하는 원소의 위치에 따라 결정
 - 검색 실패의 경우에 평균 비교 횟수가 반으로 줄어듦
- 평균 시간 복잡도 : $O(n)$

알고리즘 10-2 정렬된 자료의 순차 검색

```
sequentialSearch2(a[], n, key)
    i ← 0;
    while (a[i] < key) do {
        i ← i + 1;
    }
    if (a[i] = key) then return i;
    else return -1;
end sequentialSearch2()
```

※출처: 이지영(2016).

IT CookBook, C로 배우는 쉬운
자료구조(개정3판). 한빛미디어

3 색인 순차 검색(Index sequential search)

- ▶ 정렬되어있는 자료에 대한 인덱스 테이블(Index table)을 추가로 사용하여 탐색 효율을 높인 검색 방법
- ▶ 인덱스 테이블
 - 배열에 정렬되어있는 자료 중에서 일정한 간격으로 떨어져있는 원소들을 저장한 테이블
 - 자료가 저장되어있는 배열의 크기가 n 이고 인덱스 테이블의 크기가 m 일 때, 배열에서 n/m 간격으로 떨어져있는 원소와 그의 인덱스를 인덱스 테이블에 저장

3 색인 순차 검색(Index sequential search)

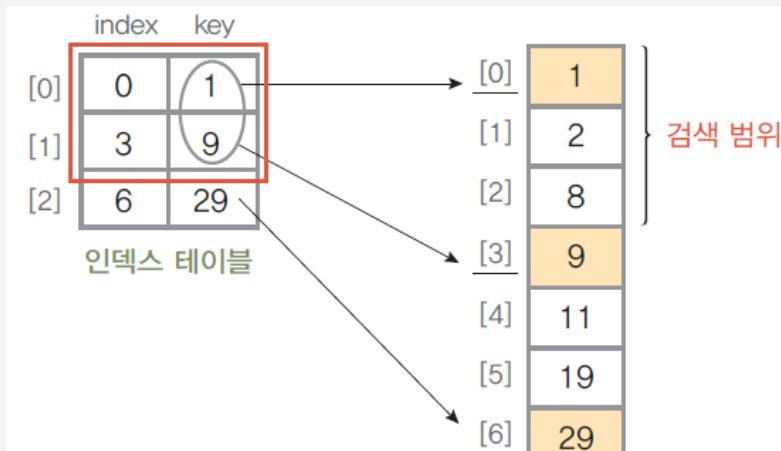
▶ 검색 방법

- $\text{indexTable}[i].key \leq \text{key} < \text{indexTable}[i+1].key$ 를 만족하는 i 를 찾아서 배열의 어느 범위에 있는지를 먼저 알아낸 후에 해당 범위에 대해서만 순차 검색 수행

4 색인 순차 검색 (예) 검색 대상 자료 : {1, 2, 8, 9, 11, 19, 29}

▶ 키값이 6인 원소를 검색하려면 먼저 인덱스 테이블의 키값을 검색

- 키값 6은 $\text{indexTable}[0].key \leq 6 < \text{indexTable}[1].key$ 로 인덱스 테이블의 키값 1과 9 사이에 있음
- 따라서 검색 범위는 $\text{indexTable}[0].index \leq \text{검색 범위} < \text{indexTable}[1].index$ 가 되므로 자료가 저장되어 있는 배열의 인덱스 0번부터 2번까지를 검색 범위로 정하고 순차 검색을 수행



[색인 순차 검색의 예]

※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

5 색인 순차 검색 알고리즘

알고리즘 10-3 정렬된 자료의 색인 순차 검색

```
indexSearch(a[], n, key)
    for (i ← 0; i < m; i ← i+1) do
        if ((indexTable[i].key ≤ key) and (indexTable[i + 1].key > key)) then {
            begin ← indexTable[i].index;
            end ← indexTable[i + 1].index;
            break;
        }
        sequentialSearch2(a[], begin, end, key);
    end indexSearch()
```

※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

5 색인 순차 검색 알고리즘

▶ 색인 순차 검색을 위해 인덱스 테이블 알고리즘

알고리즘 10-4 인덱스 테이블

```
makeIndexTable(a[], n, key)
    n ← size / m;                                // 인덱스 테이블에 들어가는 배열 원소의 간격 계산
    if (size mod m > 0) then n ← n + 1;
    for (i ← 0; i < m; i ← i+1) do {           // 인덱스 테이블 채우기
        indexTable[i].index ← i*n;
        indexTable[i].key ← a[i*n];
    }
end makeIndexTable()
```

※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

6 색인 순차 검색의 성능

- ▶ 인덱스 테이블의 크기에 따라 결정
 - 인덱스 테이블의 크기가 줄어들면 배열의 인덱스를 저장하는 간격이 커지므로 배열에서 검색해야 하는 범위도 커짐
 - 인덱스 테이블의 크기를 늘리면 배열의 인덱스를 저장하는 간격이 작아지므로 배열에서 검색해야 하는 범위는 작아지겠지만 인덱스 테이블을 검색하는 시간이 늘어나게 됨
- ▶ 색인 순차 검색의 시간 복잡도 : $O(m + n/m)$
 - 배열의 크기 : n
 - 인덱스 테이블의 크기 : m

3 | 이진 검색 방법과 이진 트리 검색 방법

1 이진 검색(Binary search), 이분 검색, 보간 검색(Interpolation search)

- ▶ 자료의 가운데에 있는 항목을 키 값과 비교하여 다음 검색 위치를 결정하여 검색을 계속하는 방법
 - 찾는 키 값 > 원소의 키 값
: 오른쪽 부분에 대해서 검색 실행
 - 찾는 키 값 < 원소의 키 값
: 왼쪽 부분에 대해서 검색 실행
- ▶ 키를 찾을 때까지 이진 검색을 순환적으로 반복 수행함으로써 검색 범위를 반으로 줄여가면서 더 빠르게 검색

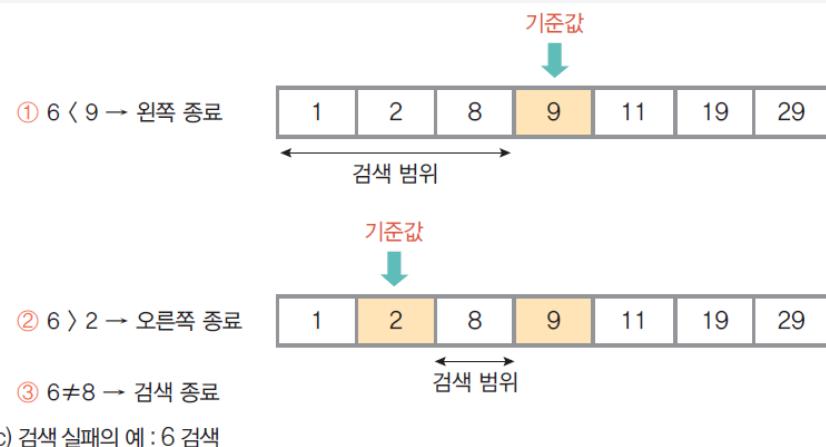
1 이진 검색(Binary search), 이분 검색, 보간 검색(Interpolation search)

- ▶ 정복 기법을 이용한 검색 방법
 - 검색 범위를 반으로 분할하는 작업과 검색 작업을 반복 수행
- ▶ 정렬되어있는 자료에 대해서 수행하는 검색 방법

3 | 이진 검색 방법과 이진 트리 검색 방법

1 이진 검색(Binary search), 이분 검색, 보간 검색(Interpolation search)

(a) 이진 검색 자료 예



[검색 실패의 예 : 6 검색]

※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

2 이진 검색 알고리즘

알고리즘 10-5 이진 검색

```
binarySearch(a[], begin, end, key)
    middle ←(begin + end) / 2;
    if (key = a[middle]) then return  1;
    else if (key < a[middle]) then binarySearch(a[], begin, middle - 1, key);
    else if (key > a[middle]) then binarySearch(a[], middle + 1, end, key);
    else return -1;
end binarySearch()
```

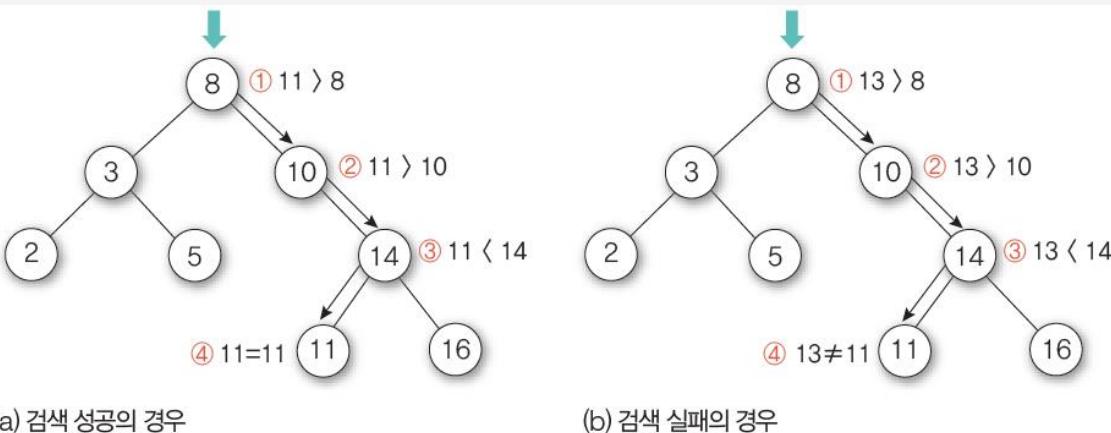
- ▶ n개의 자료에 대한 이진 검색의 메모리 사용량은 n이 됨
- ▶ 이진 검색에서 검색 범위를 1/2로 분할하면서 비교하는 연산에 대한 시간 복잡도는 $O(\log_2 n)$

※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

3 이진 트리 검색(Binary tree search)

▶ 7장에서 설명한 이진 탐색 트리를 사용한 검색 방법

[이진 트리 검색의 예
: 11검색과 13검색]

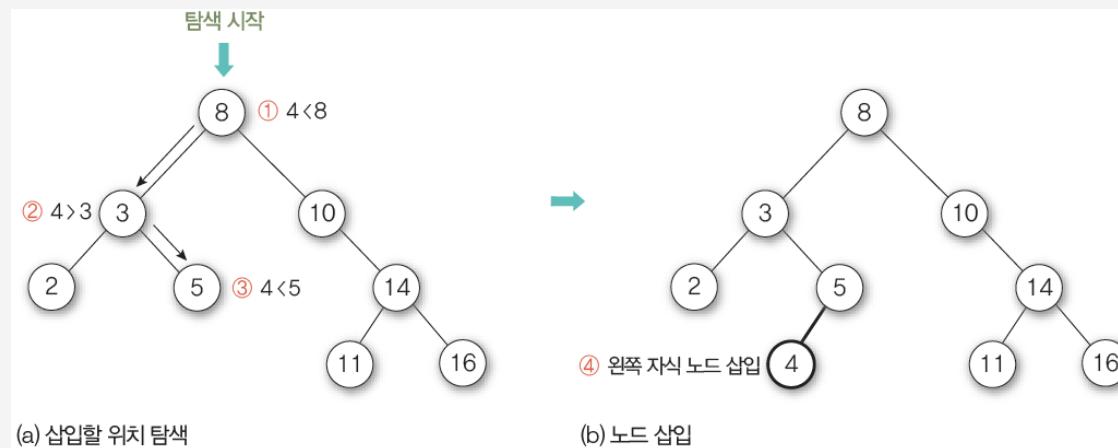


※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

3 이진 트리 검색(Binary tree search)

▶ 원소의 삽입이나 삭제 연산에 대해서 항상 이진 탐색 트리를 재구성하는 작업 필요

[이진 탐색 트리에 원소 4를 삽입 하기 : 탐색 실패한 자리에 삽입 됨]



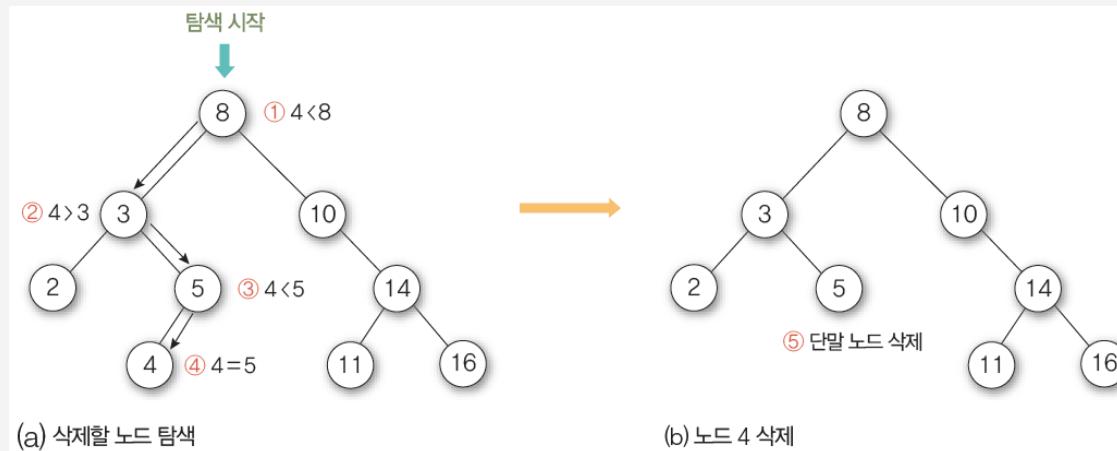
※ 출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어



3 이진 트리 검색(Binary tree search)

▶ 원소의 삽입이나 삭제 연산에 대해서 항상 이진 탐색 트리를 재구성하는 작업 필요

[이진 탐색 트리에 원소 4를 삭제 하기 : 탐색이 성공하면 삭제 가능]



※ 출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어