1 셀 정렬

1 셸 정렬(Shell sort)의 이해

- 일정한 간격(Interval)으로 떨어져있는 자료들끼리 부분집합을 구성하고 각 부분집합에 있는 원소들에 대해서 삽입 정렬을 수행하는 작업을 반복하면서 전체 원소들을 정렬하는 방법
  - 전체 원소에 대해서 삽입 정렬을 수행하는 것보다 부분집합으로 나누어 정렬하게 되면 비교연산과 교환연산 감소

1 셸 정렬(Shell sort)의 이해

- 셸 정렬의 부분집합
  - 부분집합의 기준이 되는 간격을 매개변수 h에 저장
  - 한 단계가 수행될 때마다 h의 값을 감소시키고 셸 정렬을 순환 호출
    - h가 1이 될 때까지 반복
- 셸 정렬의 성능은 매개변수 h의 값에 따라 달라짐
  - 정렬할 자료의 특성에 따라 매개변수 생성 함수를 사용
  - 일반적으로 사용하는 h의 값은 원소 개수의 1/2을 사용하고 한 단계 수행될 때마다 h의 값을 반으로 감소시키면서 반복 수행

1 셸 정렬(Shell sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 셸 정렬 방법으로 정렬하는 과정

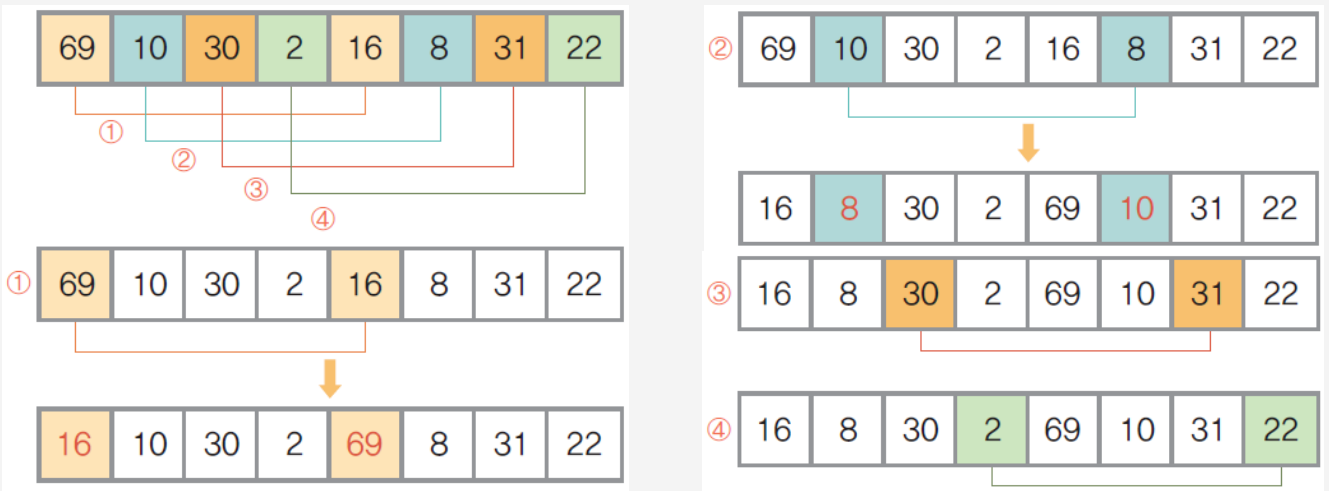
1) 원소 개수가 여덟 개이므로 매개변수 h는 4에서 시작한다. h=4이므로 간격이 4만큼 떨어져 있는 원소들을 같은 부분집합으로 만들면 네 개의 부분 집합이 만들어짐(같은 부분집합은 동일한 색으로 표시)

① 첫 번째 부분집합 {69, 16}에 대해서 삽입 정렬을 수행하여 정렬
② 두 번째 부분집합{10, 8}에 대해서 삽입 정렬을 수행
③ 세 번째 부분집합 {30, 31}에 대해서 삽입 정렬을 수행,
30<31이므로 자리 이동은 이루어지지 않음
④ 네 번째 부분집합 {2, 22}에 대해서 삽입 정렬을 수행,
2<22이므로 자리 이동은 이루어지지 않음

1 셸 정렬(Shell sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 셸 정렬 방법으로 정렬하는 과정

1)



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

1 셸 정렬(Shell sort)의 이해

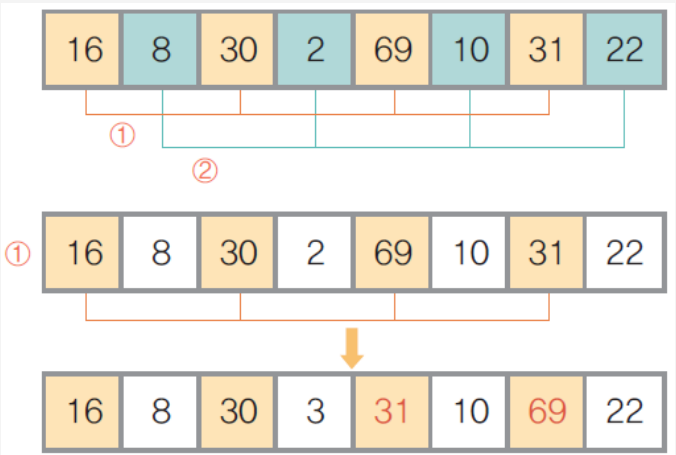
- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 셸 정렬 방법으로 정렬하는 과정

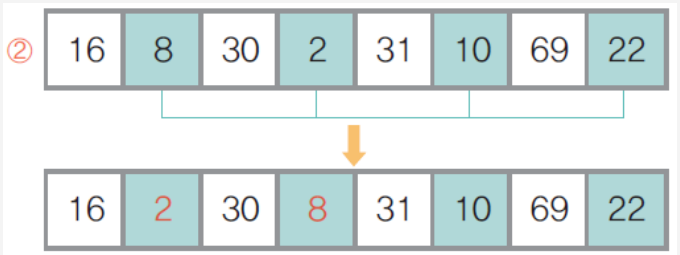
  2) 이제 h를 2로 변경하고 다시 셸 정렬을 시작. h=2이므로 간격이 2만큼 떨어진 원소들을 같은 부분집합으로 만들면 두 개의 부분집합이 만들어짐
  ① 첫 번째 부분집합 {16, 30, 69, 31}에 대해 삽입 정렬을 수행하여 정렬
  ② 두 번째 부분집합 {8, 2, 10, 22}에 대해 삽입 정렬을 수행하여 정렬

1 셸 정렬(Shell sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 셸 정렬 방법으로 정렬하는 과정

2)



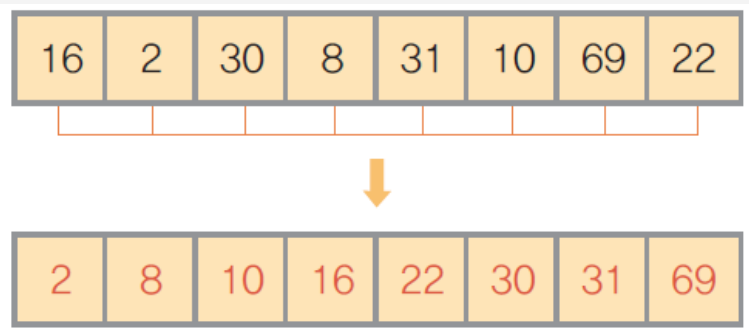


※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

1 셸 정렬(Shell sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 셸 정렬 방법으로 정렬하는 과정

  3) 이제 h를 1로 변경하고 다시 셸 정렬을 시작. h=1이므로 간격이 1만큼 떨어져 있는 원소들을 같은 부분집합으로 만들면 한 개의 부분집합이 만들어짐, 즉 전체 원소에 대해서 삽입 정렬을 수행



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

2 셸 정렬 알고리즘

알고리즘 9-6 셸 정렬

```
shellSort(a[], n)
    interval ← n;
    while (interval ≥ 1) do {
        interval ← interval / 2;
        for (i ← 0; i<interval; i←i + 1) do {
            intervalSort(a[], i, n, interval);
        }
    }
end shellSort()
```

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

2 셸 정렬 알고리즘

알고리즘 9-7 셸 부분집합에서의 삽입 정렬

```
intervalSort(a[], begin, end, interval)
for (i ← begin+interval; i≤end; i ← i+interval) do {
    item ← a[i];
    for (j ← i-interval; j≥begin and item<a[j]; j ← j-interval) do
        a[j+interval] ← a[j];
    a[j+interval] ← item;
}
end intervalSort()
```

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

2 병합 정렬

1 병합 정렬(Merge sort)의 이해

- 여러 개의 정렬된 자료의 집합을 병합하여 한 개의 정렬된 집합으로 만드는 방법
- 부분집합으로 분할(Divide)하고, 각 부분집합에 대해서 정렬 작업을 완성(Conquer)한 후에 정렬된 부분집합들을 다시 결합(Combine)하는 분할 정복(Divide and conquer) 기법 사용
- 메모리 사용공간
  - 각 단계에서 새로 병합하여 만든 부분집합을 저장할 공간이 추가로 필요

1 병합 정렬(Merge sort)의 이해

- 병합 정렬 방법의 종류
  - 2-way 병합
    : 2개의 정렬된 자료의 집합을 결합하여 하나의 집합으로 만드는 병합 방법
  - n-way 병합
    : n개의 정렬된 자료의 집합을 결합하여 하나의 집합으로 만드는 병합 방법

1 병합 정렬(Merge sort)의 이해

2-way 병합 정렬 과정

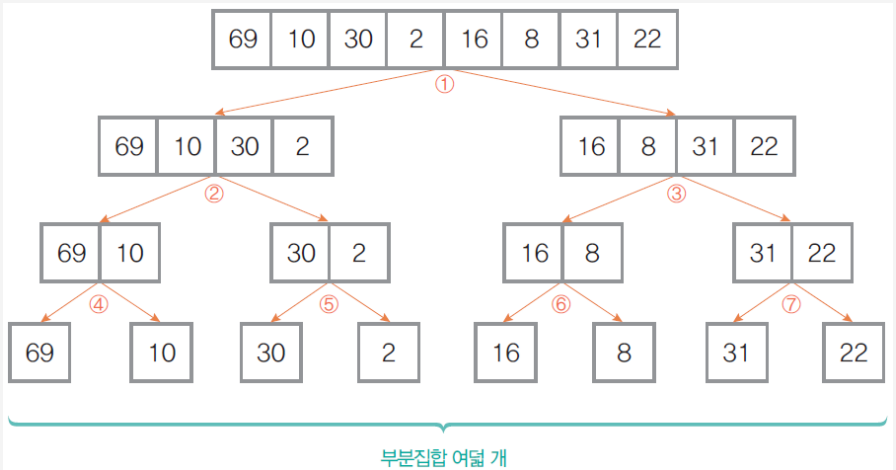
- 분할Divide : 자료들을 두 개의 부분집합으로 분할한다.
- 정복Conquer : 부분집합에 있는 원소를 정렬한다.
- 결합Combine : 정렬된 부분집합들을 하나의 집합으로 정렬하여 결합한다.

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

1 병합 정렬(Merge sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 병합 정렬 방법으로 정렬하는 과정

① 분할 단계
: 정렬할 전체 자료의 집합에 대해서 최소 원소의 부분집합이 될 때까지 분할 작업을 반복하여 한 개의 원소를 가진 부분집합 8개 만듦

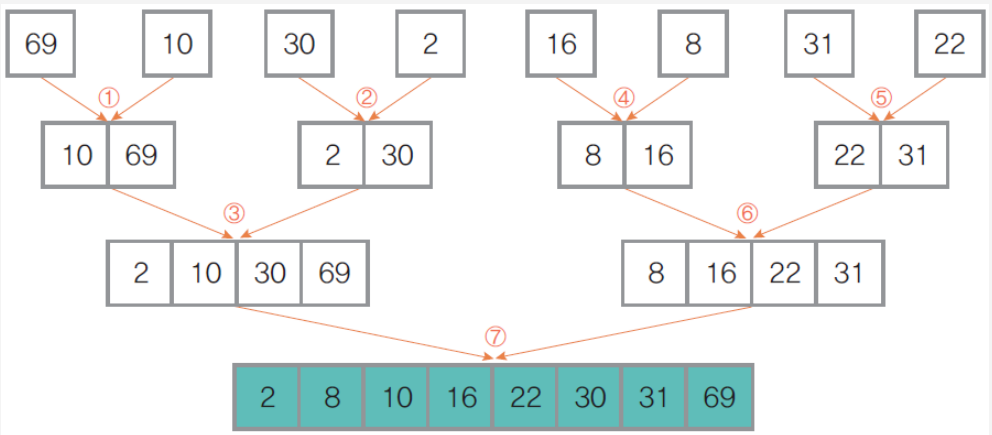


※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

1 병합 정렬(Merge sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 병합 정렬 방법으로 정렬하는 과정

② 정복과 결합 단계
: 부분집합 두 개를 정렬하여 하나로 결합, 전체 원소가 집합 하나로 묶일 때까지 반복



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

2 병합 정렬 알고리즘

알고리즘 9-8 병합 정렬

```
mergeSort(a[], m, n)
    if (a[m:n]의 원소 수 > 1) then {
        전체 집합을 부분집합 두 개로 분할;
  mergeSort(a[], m, middle);            // 왼쪽 부분집합을 다시 두 개로 분할
  mergeSort(a[], middle + 1, n);        // 오른쪽 부분집합을 다시 두 개로 분할
  merge(a[m:middle], a[middle + 1:n]);  // 부분집합 두 개를 하나로 병합
}
end mergeSort()
```

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

2 병합 정렬 알고리즘

알고리즘 9-9 부분집합의 병합

```
merge(a[m:middle], a[middle + 1:n])
  i ← m;          // 첫 번째 부분집합의 첫째 원소 설정
  j ← middle + 1; // 두 번째 부분집합의 첫째 원소 설정
  k ← m;          // 병합 집합의 첫째 원소 설정

   while (i <= middle and j <= n) do {
      if (a[i] <= a[j]) then {
         sorted[k] ← a[i];
         i ← i + 1;
      }
      else {
         sorted[k] ← a[j];
         j ← j + 1;
      }
      k ← k + 1;
   }
   if (i > middle) then 두 번째 부분집합에 남아 있는 원소를 병합 집합 sorted에 복사;
   else  첫 번째 부분집합에 남아 있는 원소를 병합 집합 sorted에 복사;
end merge()
```

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

3 히프 정렬과 트리 정렬

1 히프 정렬(Heap sort)의 이해

- 7장의 히프 자료구조를 이용한 정렬 방법

- 히프에서는 항상 가장 큰 원소가 루트 노드가 되고 삭제 연산을 수행하면 항상 루트 노드의 원소를 삭제하여 반환
  - 최대 히프에 대해서 원소의 개수만큼 삭제 연산을 수행하여 내림차순으로 정렬 수행
  - 최소 히프에 대해서 원소의 개수만큼 삭제 연산을 수행하여 오름차순으로 정렬 수행

1 히프 정렬(Heap sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 히프 정렬 방법으로 정렬하는 과정

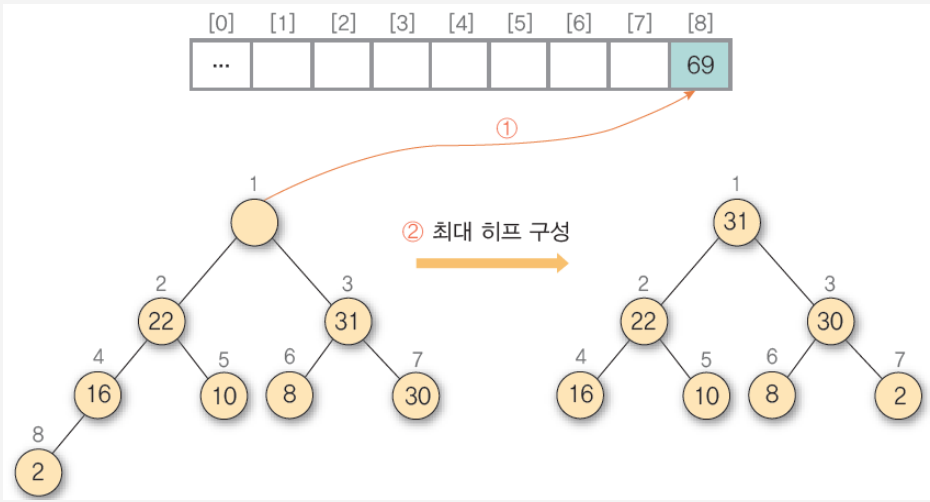
  1) 초기 상태
     : 정렬할 원소에 대해 7장에서 설명한 삽입 연산을 이용해 최대 히프를 구성

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

1 히프 정렬(Heap sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 히프 정렬 방법으로 정렬하는 과정

2) ① 히프에 삭제 연산을 수행하여 루트 노드의 원소 69를 구한 후 배열의 마지막 자리에 저장
② 나머지 히프를 최대 히프로 재구성

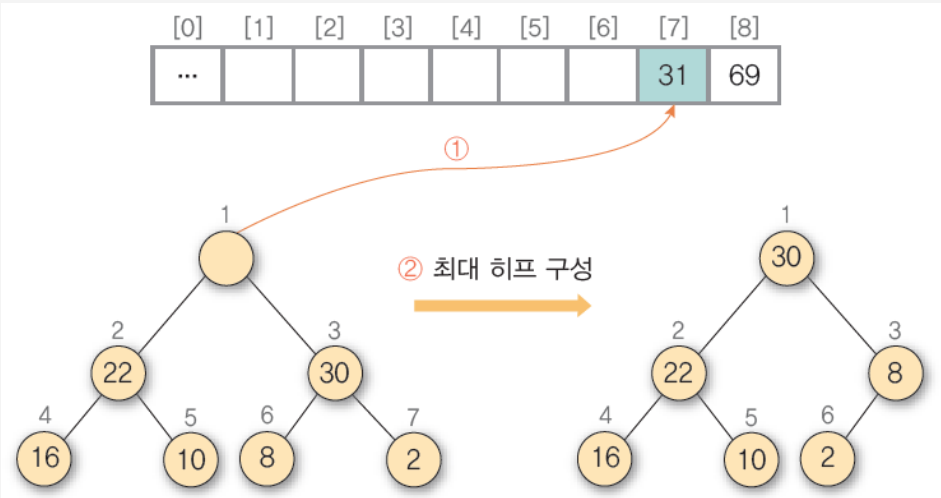


※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

1 히프 정렬(Heap sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 히프 정렬 방법으로 정렬하는 과정

3) ① 히프에 삭제 연산을 수행하여 루트 노드의 원소 31을 구한 후 배열의 비어 있는 마지막 자리에 저장
② 나머지 히프를 최대 히프로 재구성

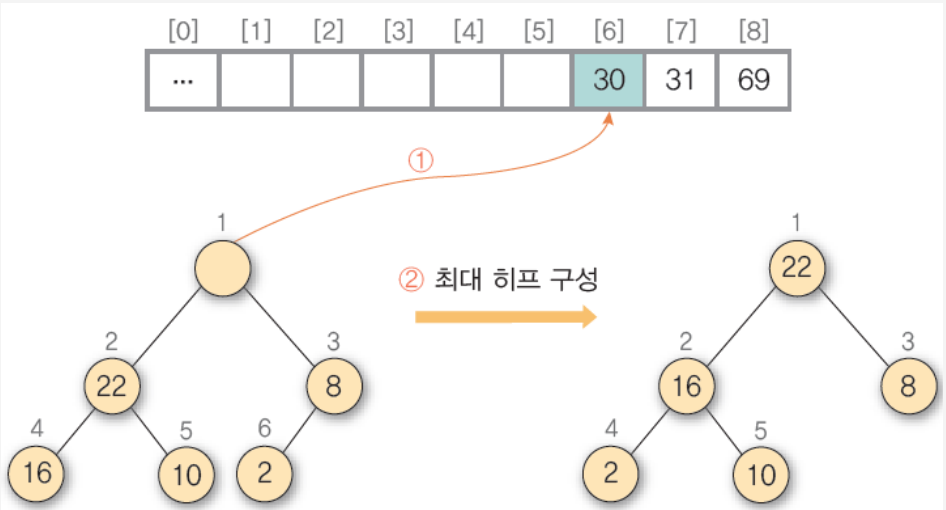


※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

1 히프 정렬(Heap sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 히프 정렬 방법으로 정렬하는 과정

4) ① 히프에 삭제 연산을 수행하여 루트 노드의 원소 30을 구한 후 배열의 비어 있는 마지막 자리에 저장
② 나머지 히프를 최대 히프로 재구성

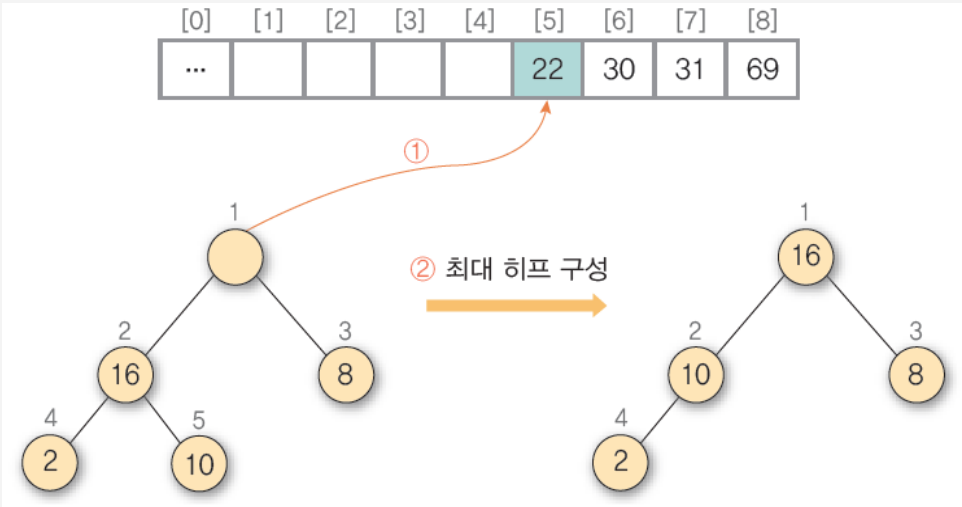


※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

1 히프 정렬(Heap sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 히프 정렬 방법으로 정렬하는 과정

5) ① 히프에 삭제 연산을 수행하여 루트 노드의 원소 22를 구한 후 배열의 비어 있는 마지막 자리에 저장
② 나머지 히프를 최대 히프로 재구성

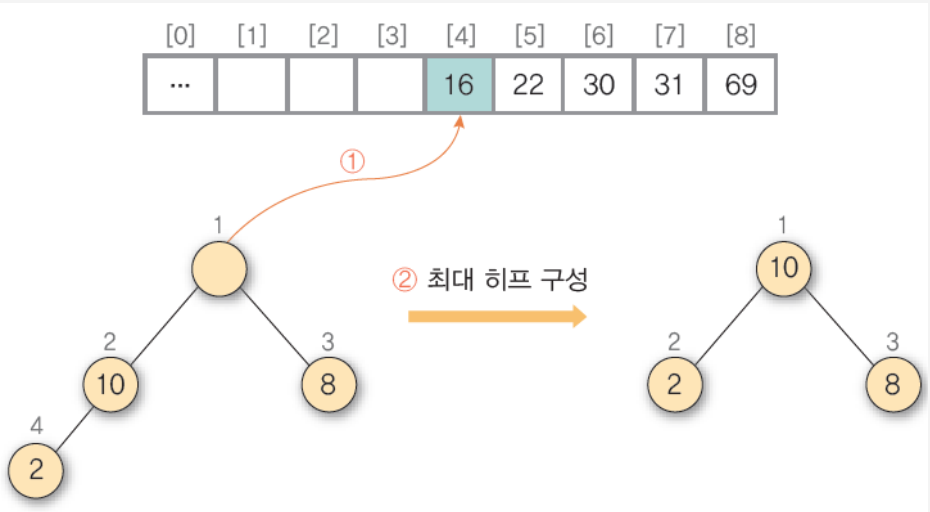


※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

1 히프 정렬(Heap sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 히프 정렬 방법으로 정렬하는 과정

6) ① 히프에 삭제 연산을 수행하여 루트 노드의 원소 16을 구한 후 배열의 비어 있는 마지막 자리에 저장
② 나머지 히프를 최대 히프로 재구성

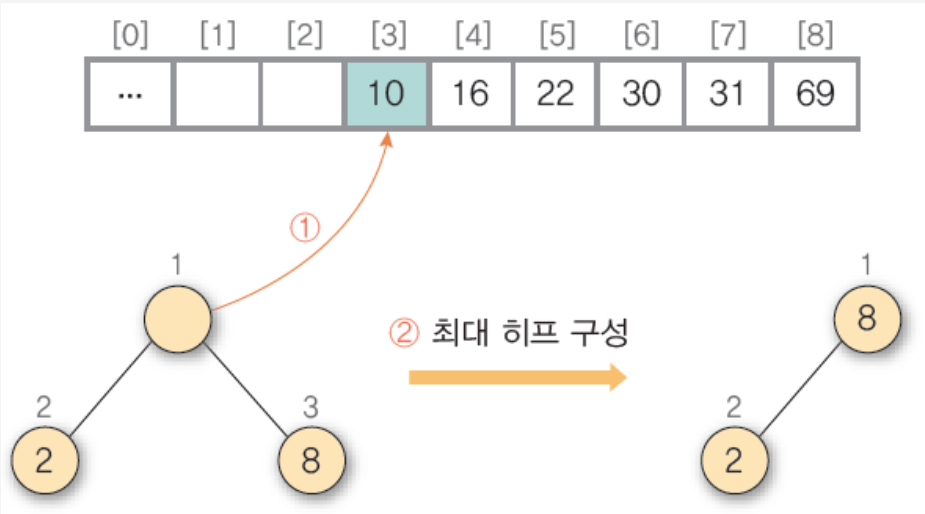


※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

1 히프 정렬(Heap sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 히프 정렬 방법으로 정렬하는 과정

7) ① 히프에 삭제 연산을 수행하여 루트 노드의 원소 10을 구한 후 배열의 비어 있는 마지막 자리에 저장
② 나머지 히프를 최대 히프로 재구성

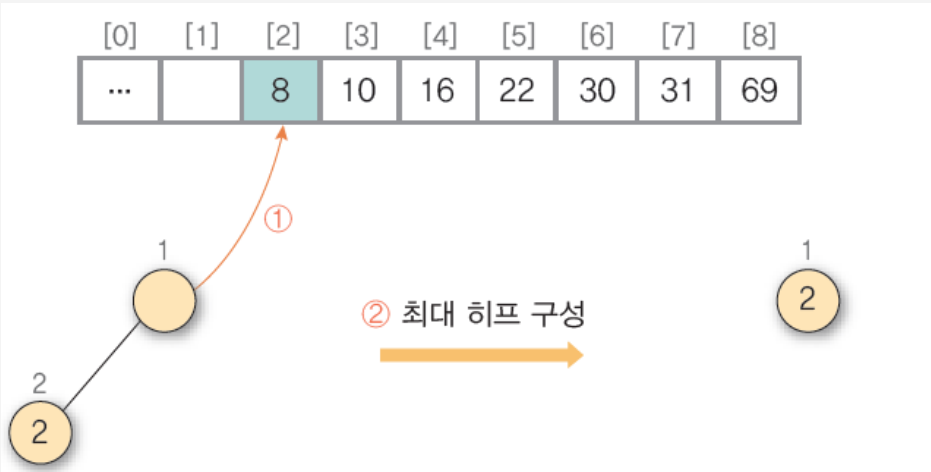


※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

1 히프 정렬(Heap sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 히프 정렬 방법으로 정렬하는 과정

8) ① 히프에 삭제 연산을 수행하여 루트 노드의 원소 8을 구한 후 배열의 비어 있는 마지막 자리에 저장
② 나머지 히프를 최대 히프로 재구성

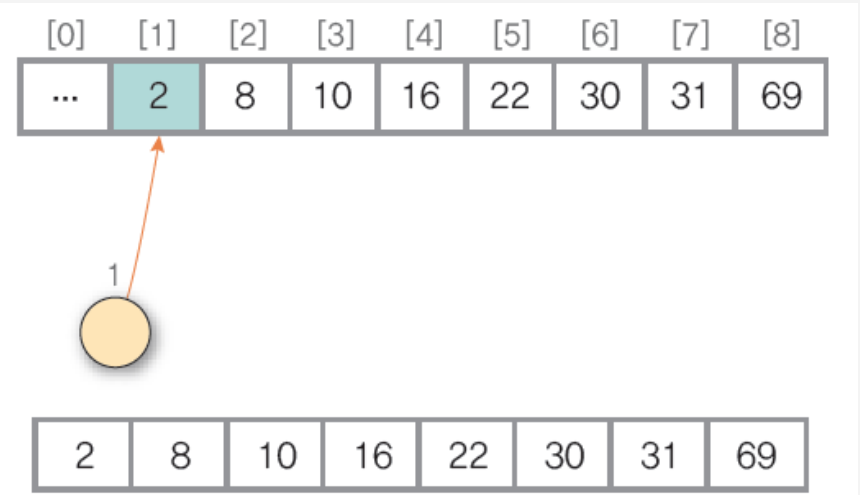


※출처: 이지영(2016). IT CookBook, C로 배우는 쉬운 자료구조(개정3판). 한빛미디어

1 히프 정렬(Heap sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 히프 정렬 방법으로 정렬하는 과정

9) 히프에 삭제 연산을 수행하여 루트 노드의 원소 2를 구한 후 배열의 비어 있는 마지막 자리에 저장, 나머지 히프를 최대 히프로 재구성해야 하는데 공백 히프가 되었으므로 히프 정렬을 종료



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

2 히프 정렬 알고리즘

알고리즘 9-11 히프 정렬

```
heapSort(a[])
    n ← a.length - 1;
    for (i ← n / 2; i ≥ 1; i ← i - 1) do  // 배열 a[]를 히프로 변환
        makeHeap(a, i, n);
    for (i ← n - 1; i ≥ 1; i ← i - 1) do {
        temp ← a[1];     // 히프의 루트 노드 원소를
        a[1] ← a[i + 1]; // 배열의 비어 있는
        a[i + 1] ← temp; // 마지막 자리에 저장
        makeHeap(a, 1, i);
    }
end heapSort()
```

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

2 히프 정렬 알고리즘

- 배열을 히프 구조로 재구성하는 makeHeap( ) 연산에 대한 알고리즘

알고리즘 9-12 히프 재구성

```
makeHeap(a[], h, m)
    for (j ← 2 * h; j ≤ m; j ← 2 * j) do {
        if (j < m) then
            if (a[j] < a[j + 1]) then j ← j + 1;
        if (a[h] ≥ a[j]) then exit;
        else a[j / 2] ← a[j];
    }
    a[j / 2] ← a[h];
end makeHeap()
```

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

3 트리 정렬(Tree sort)의 이해

- 7장의 이진 탐색 트리를 이용하여 정렬하는 방법
- 트리 정렬 수행 방법
  - 정렬할 원소들을 이진 탐색 트리로 구성
  - 이진 탐색 트리를 중위 우선 순회 함
    - 중위 순회 경로가 오름차순 정렬이 됨

3 트리 정렬(Tree sort)의 이해

- 정렬되지 않은 {69, 10, 30, 2, 16, 8, 31, 22}의 자료들을 트리 정렬 방법으로 정렬하는 과정
  - ① 정렬할 원소 여덟 개를 차례대로 삽입하여 이진 탐색 트리를 구성
  - ② 이진 탐색 트리를 중위 순회 방법으로 순회하면서 원소를 저장

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

4 트리 정렬 알고리즘

- 이진 탐색 트리에서의 삽입 연산 insert( )와 중위 순회 연산 inorder( )는 7장 알고리즘을 응용하여 사용함

알고리즘 9-13 트리 정렬

```
treeSort(a[], n)
    for (i ← 0; i < n; i ← i + 1) do
        insert(BST, a[i]);      // 이진 탐색 트리의 삽입 연산
    inorder(BST);               // 중위 순회 연산
end treeSort()
```

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

5 정렬 알고리즘 성능 비교

데이터 n개일 경우 각 정렬 알고리즘 시간 복잡도

알고리즘	최선	평균	최악
삽입 정렬	$O(n)$	$O(n^2)$	$O(n^2)$
선택 정렬	$O(n^2)$	$O(n^2)$	$O(n^2)$
버블 정렬	$O(n^2)$	$O(n^2)$	$O(n^2)$
쉘 정렬	$O(n)$	$O(n^{1.5})$	$O(n^{1.5})$
퀵 정렬	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n^2)$
히프 정렬	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$
병합 정렬	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$

※출처: 천인국(2014). C 언어로 쉽게 풀어 쓴 자료구조(개정2판). 생능출판사

5 정렬 알고리즘 성능 비교

▶ 정수 6000개 데이터 정렬 알고리즘 실행 시간

알고리즘	실행 시간(단위:sec)
삽입 정렬	7.438
선택 정렬	10.842
버블 정렬	22.894
쉘 정렬	0.056
히프 정렬	0.034
합병 정렬	0.026
퀵 정렬	0.014

※출처: 천인국(2014). C 언어로 쉽게 풀어 쓴 자료구조(개정2판). 생능출판사