히프와 우선순위 큐
Heap & Priority Queue

만든이 서정현
작성일 2017.09.27

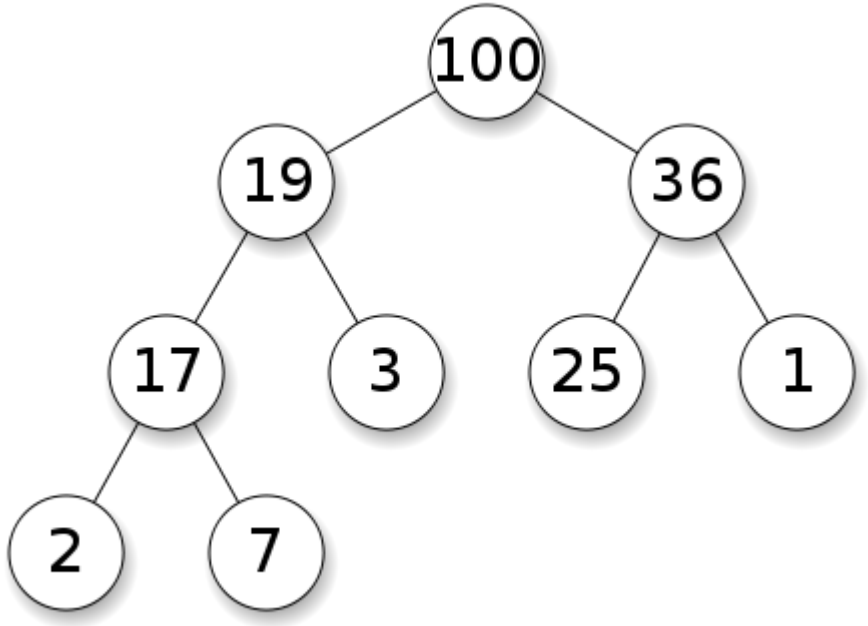
01. Heap

특정한 규칙을 만족하도록 구성된 **완전 이진 트리** (Complete Binary Tree)

- 부모 노드의 키 값과 자식 노드의 키 값 사이에는 대소 관계가 성립

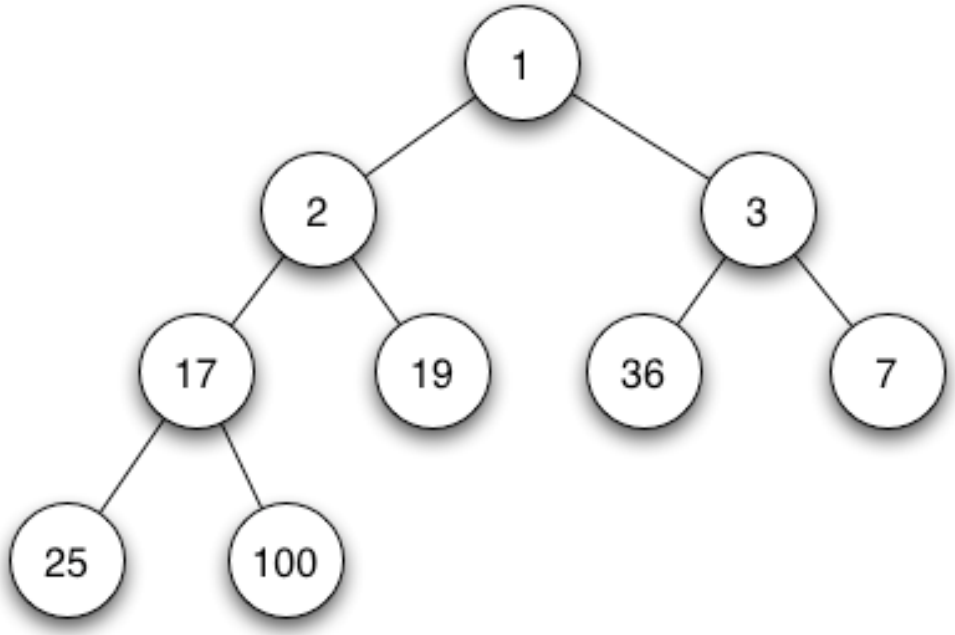
Max Heap

부모 노드의 키 값이 자식 노드의 키 값보다 항상 **큰** 힙



Min Heap

부모 노드의 키 값이 자식 노드의 키 값보다 항상 **작은** 힙



01. Heap?

- Binary Heap의 특징

- 키 값의 대소 관계는 오로지 부모 노드와 자식 노드 간에만 성립
- 형제 사이에는 대소 관계가 정해지지 않는다.

- 자식 노드의 개수가 최대 2개 -> Binary heap
- 시간 복잡도: $O(\log n)$

- Binomial heap, Fibonacci heap

02. 우선순위 큐

- 큐나 스택과 같은 축약 자료형으로, 각 원소들은 우선순위를 가진다.

- 높은 우선순위를 가진 원소는 낮은 우선순위를 가진 원소보다 먼저 처리
- 두 원소가 같은 우선순위를 가진다면, 큐에서 그들의 순서에 의해 처리

- 우선순위 큐는 heap이 아니다.
  - 리스트나 맵과 같은 추상적인 개념
  - 다양한 방법을 이용해 구현 가능

- 응용 분야
  - 시뮬레이션 시스템
  - 네트워크 트래픽 제어
  - 운영체제에서의 작업 스케쥴링

02. 우선순위 큐

–구현 방법에 따른 성능 비교

- **배열**

- 데이터 삽입 및 삭제 과정에서 데이터를 한 칸씩 당기거나 밀어야 하는 연산을 계속해야 한다.
- 삽입의 위치를 찾기 위해 배열에 저장된 모든 데이터와 우선순위를 비교해야 한다. (정렬되지 않은 배열)

- **연결 리스트**

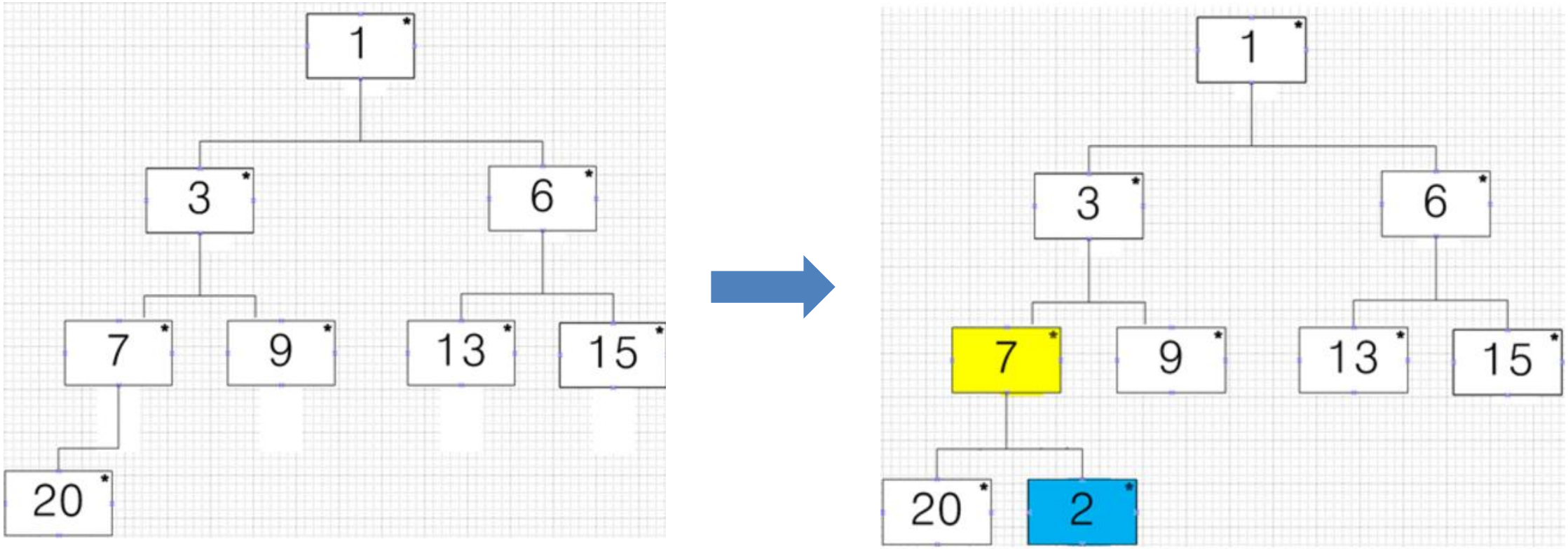
- 삽입의 위치를 찾기 위해 첫번째 노드에서부터 시작해 마지막 노드에 저장된 데이터와 우선순위 비교를 진행하는 경우가 생김
- 데이터가 많아질 때 노드의 수에 비례해서 비교할 대상이 증가 -> 성능 저하

표현 방법	삽 입	삭 제
정렬되지 않은 배열	O(1)	O(n)
정렬되지 않은 연결 리스트	O(1)	O(n)
정렬된 배열	O(n)	O(1)
정렬된 연결 리스트	O(n)	O(1)
히프	O(lg(n))	O(lg(n))

그래서, **Heap** 좋음

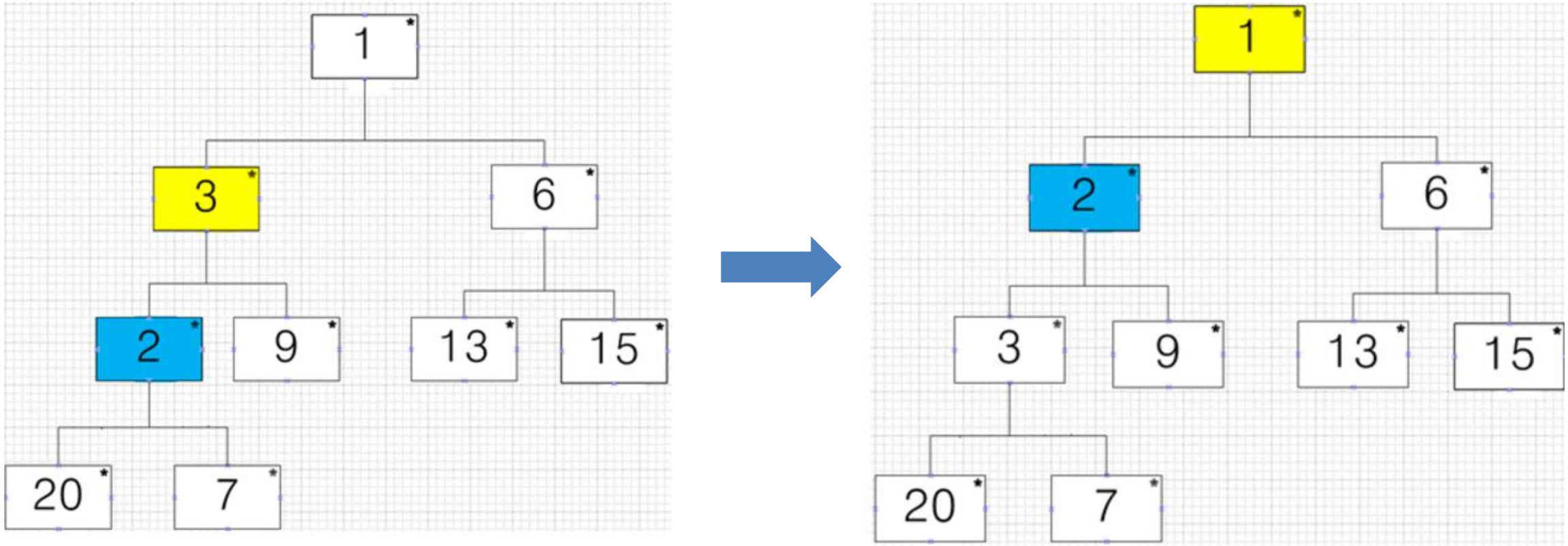
03. 힙 우선순위 큐

- 삽입



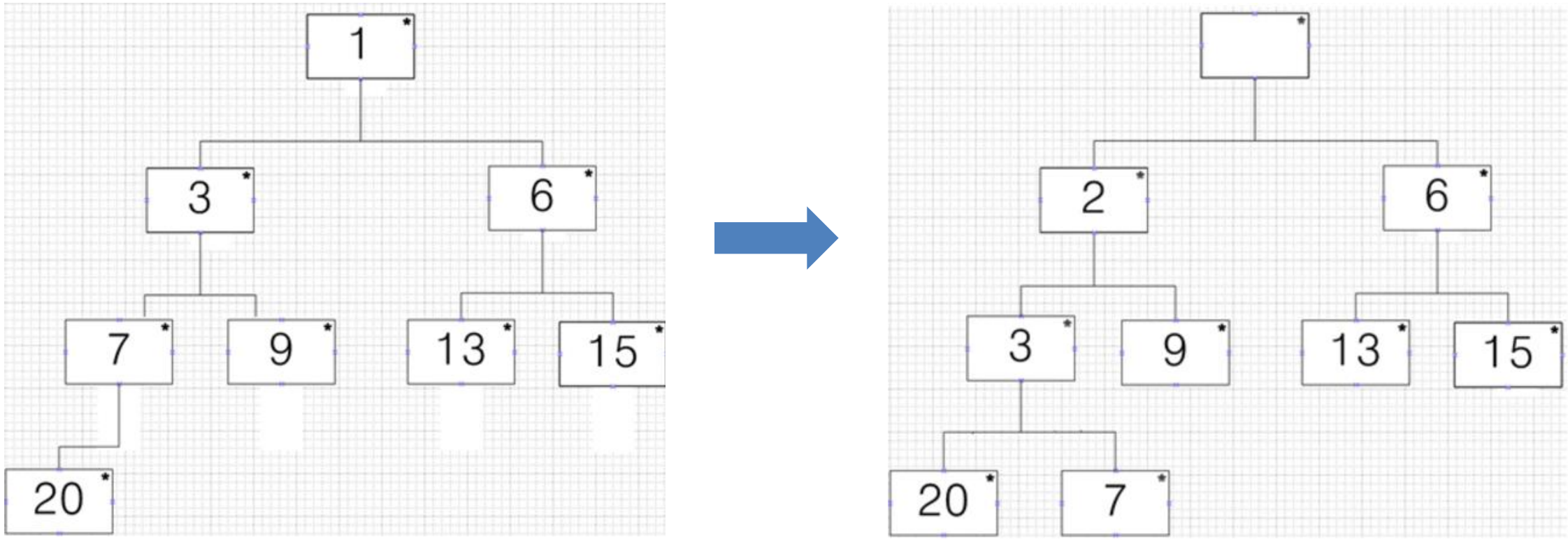
03. 힙 우선순위 큐

- 삽입



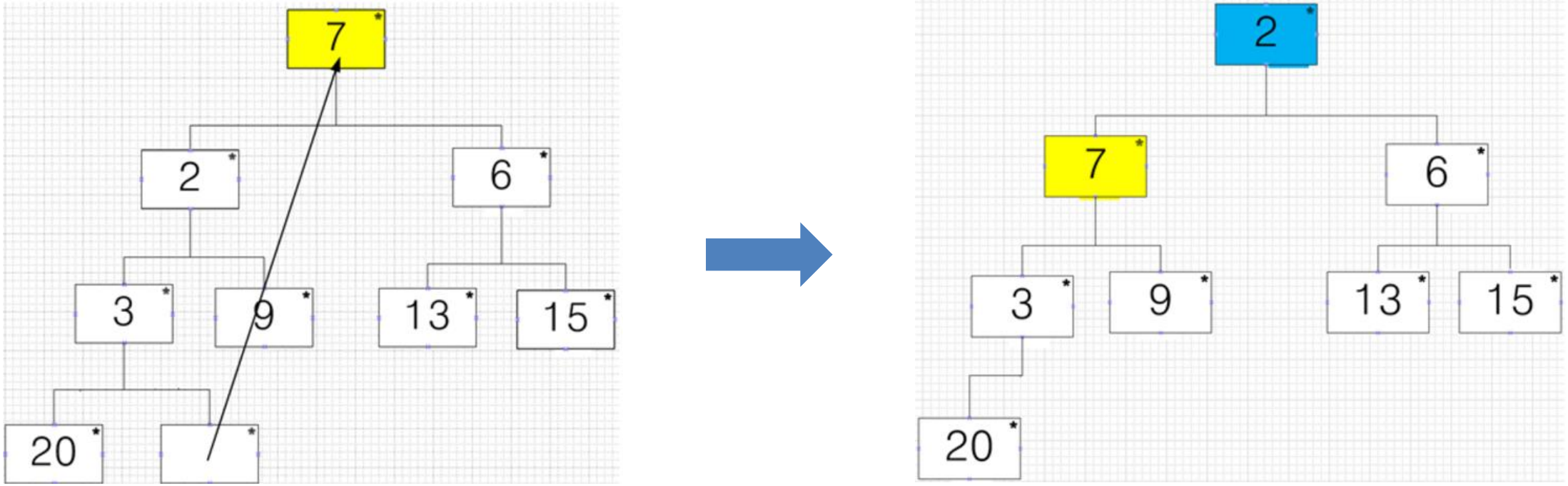
03. 힙 우선순위 큐

- 삭제



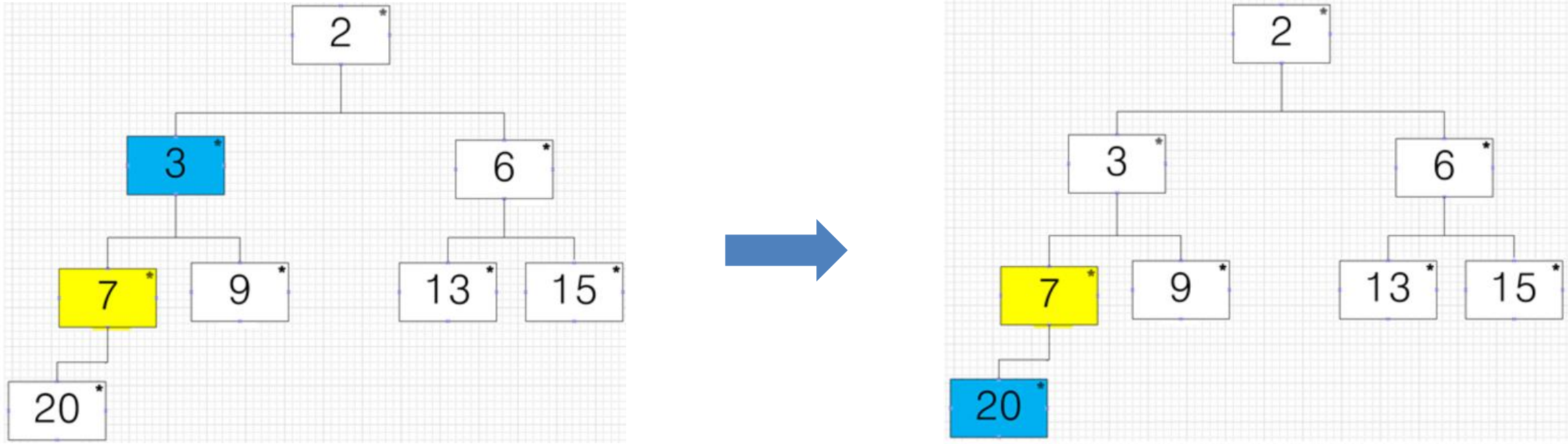
03. 힙 우선순위 큐

- 삭제



03. 힙 우선순위 큐

- 삭제



04. 허프만 코드

- 각 글자의 빈도 수를 기반으로 메시지를 압축을 위해 고안된 특수화된 코드 형태

- 빈도가 높은 정보- 적은 비트 수
- 빈도가 낮은 정보- 많은 비트 수

- 전체 데이터의 표현에 필요한 비트의 양을 줄임

- 가변 길이 코드,영상 압축에 많이 사용
- 팩스, 모뎀, 컴퓨터 네트워크, 고해상도 텔레비전

문자	코드
a	000
b	001
c	010
d	011
e	100
f	101
g	110
h	111

표 2) 압축을 반영하지 않은 코드

문자 비트 수가 줄어든다.

문자	코드
a	11
b	00
c	101
d	010
e	1000
f	1001
g	0110
h	0111

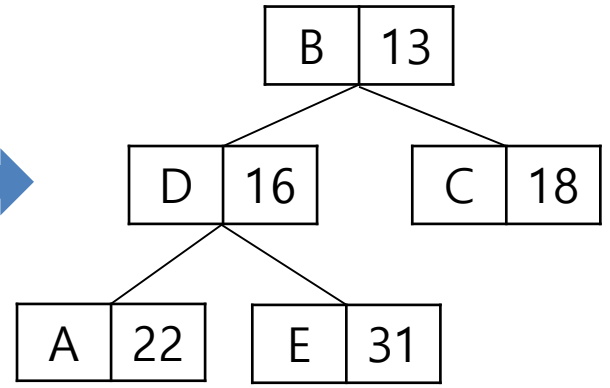
표 3) 허프만 코드

04. 허프만 코드

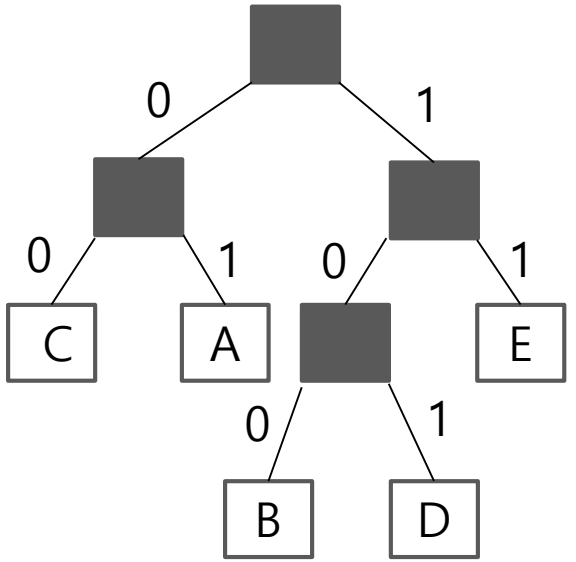
- 테스트

문자	빈도수
A	22%
B	13%
C	18%
D	16%
E	31%

최소 히프



허프만 트리



허프만 코드

문자	코드
A	01
B	100
C	00
D	101
E	11

하나의 문자 코드가 다른 어떤 문자 코드의
접두부(prefix)와도 겹치지 않는다
→ 코딩의 유일성 보장