

그리디 알고리즘

1/22

SCCC 정주영 @greimul

그리디 알고
리즘이란?

Greedy: 탐욕스러운, 욕심많은



말그대로 그때 그때 최선의 이익을
선택하는 알고리즘 입니다.

그리디 알고리즘의 조건

- Greedy choice property
 - -앞의 선택이 이후의 선택에 영향을 미치지 않는다.
 - (한번 선택한 것은 다시 고려하지 않는다.)
- Optimal substructure
 - -문제에 대한 최적해가 부분문제에 대해서도 최적해 이다.

그리디 알고리즘의 조건

EX) n 개의 물건이 있고 각각의 물건에 p_n 라는 가치가 주어질 때, m 개를 골라서 최대 이익을 만들어라.

- Greedy choice property

- 앞의 선택이 이후의 선택에 영향을 미치지 않는다.

- >한 물건을 집었다고 다른 물건을 못 집는 것은 아님.

- Optimal substructure

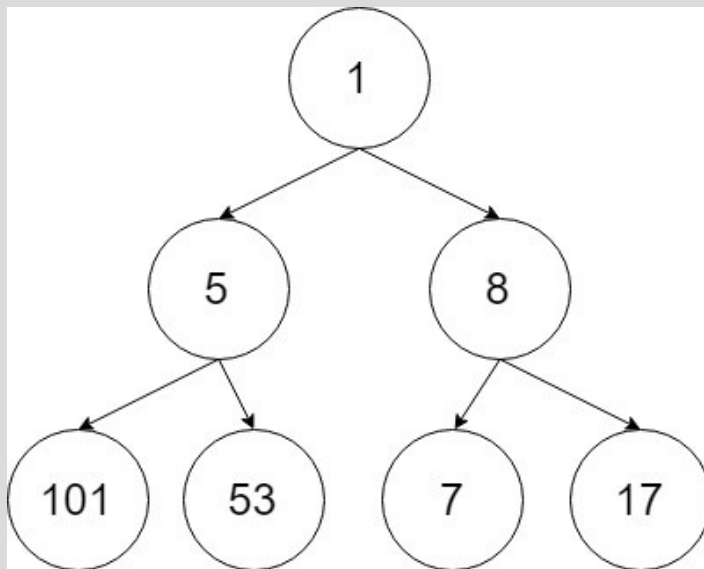
- 문제에 대한 최적해가 부분문제에 대해서도 최적해 이다.

- > 직관이 중요한 부분.

- > 가장 많은 이득을 취하려면 당연히 그때그때 가치가 가장 큰 물건을 골라야 한다.

그리디 알고리즘의 조건

- EX) 가장 합이 큰 path를 구하라.



그리디 알고리즘

$1 \rightarrow 8 \rightarrow 17$

$= 26$

실제 답

$1 \rightarrow 5 \rightarrow 101$

$= 107$

한번의 선택이 이후 선택에 영향을 준다.

부분에서 이득을 취하는 것이 꼭 답이 아니다.

그러므로 그리디로는 해결 불가능.

대표적인 그리디 알고리즘 문제

1. Fractional Knapsack Problem / 거스름돈 문제

2. 시간표 배정

3. 수학적 성질이 최적해를 보장하는 경우

4. Huffman Code

5. 다익스트라

6. MST (Kruskal, Prim)

7. 최대 유량

8. "리듬게임"도 그리디 문제였다!



출석체크 겸 문제풀기!

동전 0
11047번

동전 0


- 구하고자 하는 것: K원을 만들기 위해 필요한 동전의 최소 개수

$A_1 = 1, i \geq 2$ 인 경우에 A_i 는 A_{i-1} 의 배수

- 위 조건 덕분에 답은 무조건 나옴

동전 0

- 결론은?
- 가치가 큰 동전 부터 넣으면 된다!

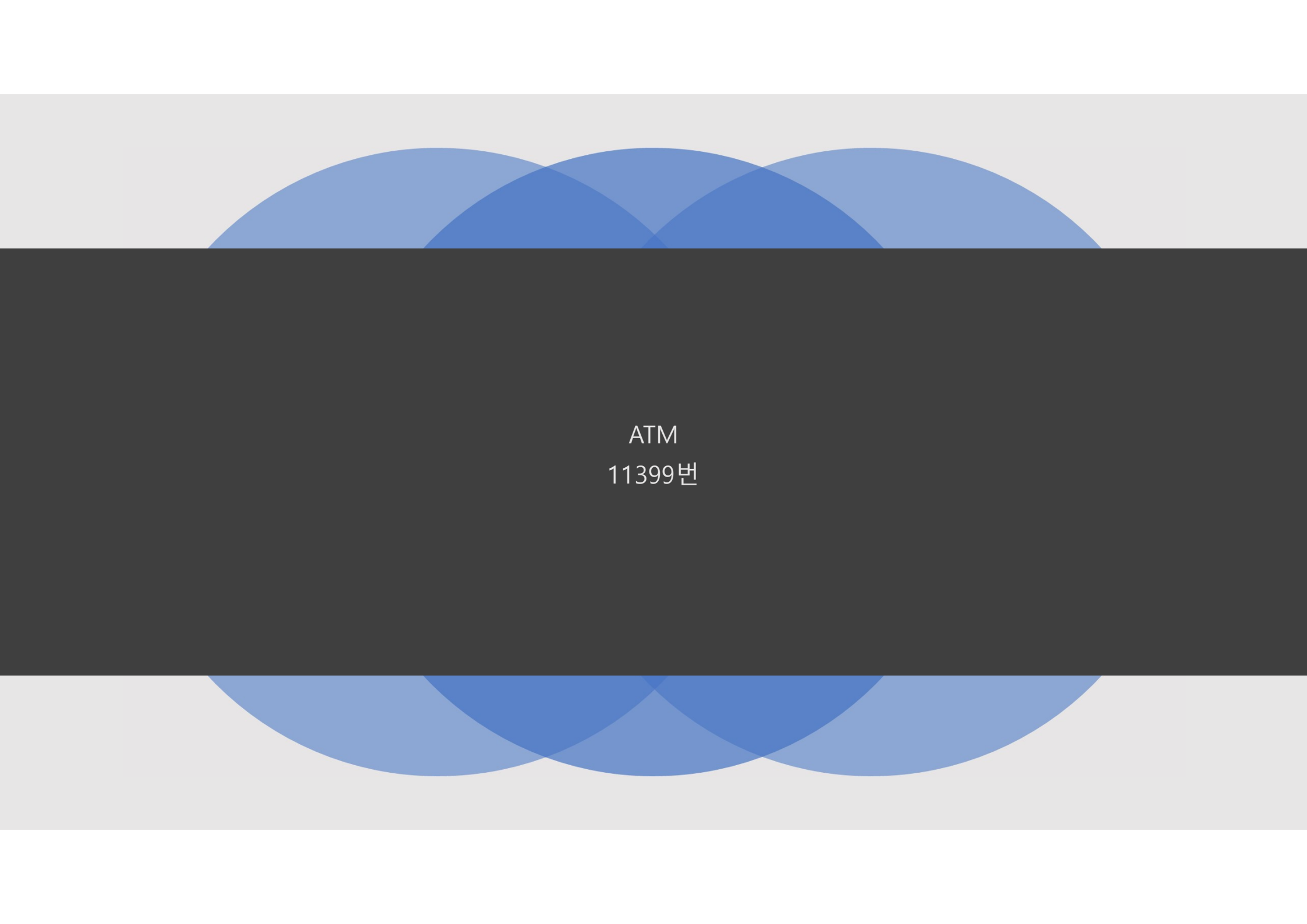


시간 배정

회의실 배정
1931번

회의실 배정

- 끝나는 시간으로 정렬한 후, 가장 빨리 끝나는 회의부터 선택하면 해결.
- 회의실 배정문제는 다양하게 변형해서 출제 가능하다.



ATM
11399번

ATM

- 3명이 있을 때,
- $x + (x + y) + (x + y + z) = 3x + 2y + z$
- N명이 있을 때,
- $nx_1 + (n - 1)x_2 + (n - 2)x_3 + \cdots + x_n$
- 숫자들을 정렬한 이후, 순서대로 식에 대입해주면 해결!

Huffman Code

- 문자열 압축 알고리즘
- 실제로 압축에서 많이 쓰이는 알고리즘이다.
- 자주 보이진 않지만 중요한 알고리즘
- 쉬운 문제에서는 직접적으로 물어볼 수 있지만,
- "가장 작은 두 원소들을 합쳐라 " 라는 식의 문제로 간접 출제 가능

Huffman Code

- 스트링 ABABCBBBC 가 있다고 가정하자. (9bytes = 72bits)
- A: 00, B: 01, C: 11
- 000100011101010111 - 18bits
- 각 문자를 대표하는 codeword의 길이가 같은 경우
- Fixed-length binary code!
- 상황에 맞춰서 다른 길이도 사용할 수 있으면 어떨까?
- 두 자리 대신 한자리 길이를 이용한다면?

Huffman Code

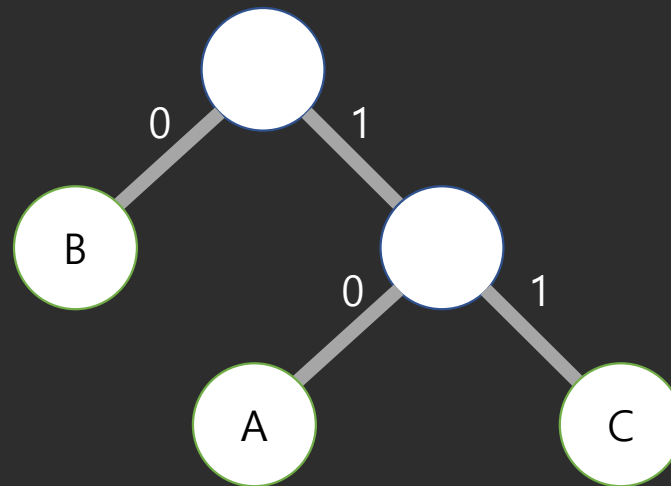
- 빈도수를 따져보자!
- A: 2번, B: 5번, C: 2번
- 빈도수가 높은 것에 길이가 짧은 codeword를 할당해주자
- B: 0, A: 10, C: 11
- 1001001100011 – 13bits!
- Variable-Length binary code!

Huffman Code

- Huffman code 에서 쓰는 것은 variable-length code 중 하나인
- Prefix code!
- Prefix code?
- 어떠한 codeword도 다른 문자에 해당하는 codeword의 시작부분이 될 수 없다.
- Ex) 01이 codeword로 존재하면 011 은 있을 수 없다.
- 001이 있으면 00이 있을 수 없다.
- Fixed-length code 도 Prefix code에 속한다.

Huffman Code

- Prefix Code는 바이너리 트리로 표현 가능하다.
- EX)



- B: 0, A: 10, C: 11
- ABABCBBBC 1001001100011

Huffman Code

- Huffman code의 목표는 가장 짧은 비트로 압축하는 것.
- 빈도수가 더 적은 문자를 더 짧은 문자열을 할당한다.
- Priority_queue 를 이용한 그리디 알고리즘으로 해결 가능하다.

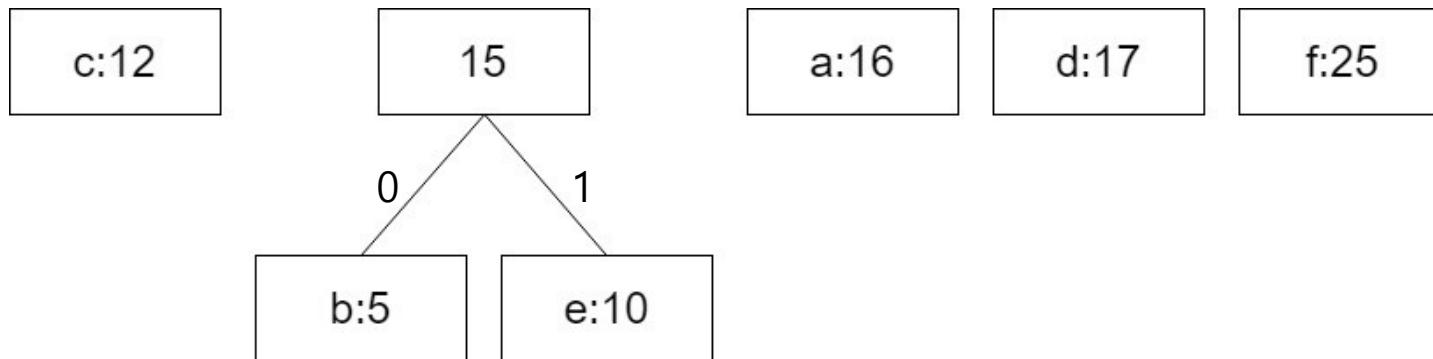
Huffman Code

- 각 문자의 빈도수를 구한다.
- EX) a: 16, b: 5, c:12, d: 17, e: 10 f: 25 인 문자열
- 각 빈도수를 PQ에 모두 넣는다.
- PQ에서 두 개씩 빼면서 가장 작은 두개의 수를 찾아서 합친다.

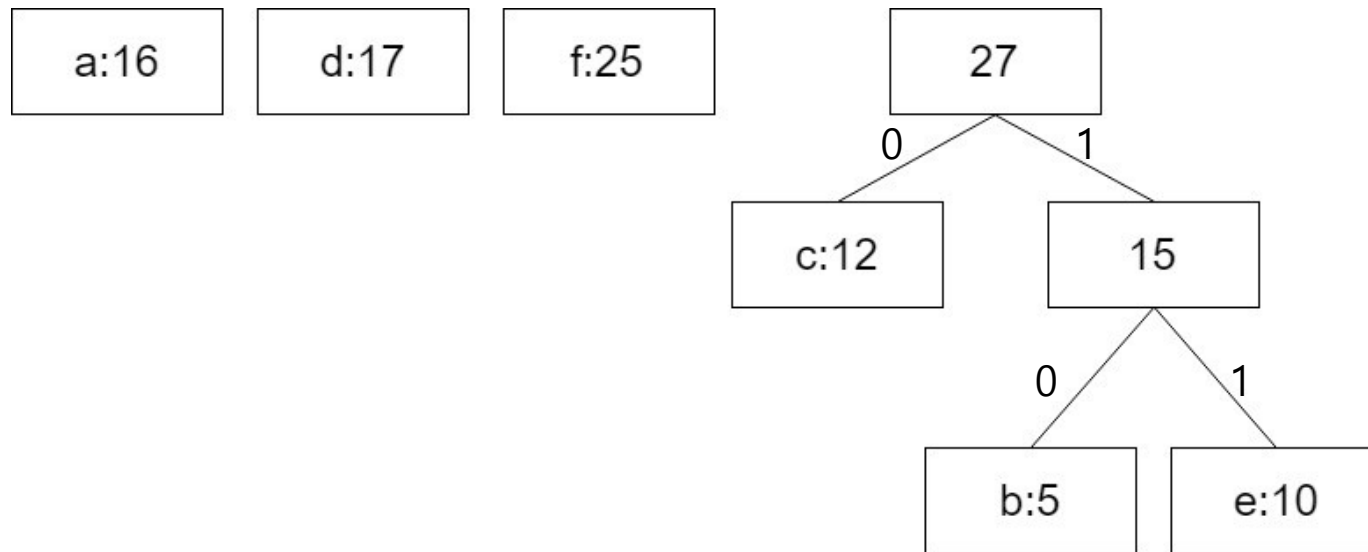
Huffman Code

b:5	e:10	c:12	a:16	d:17	f:25
-----	------	------	------	------	------

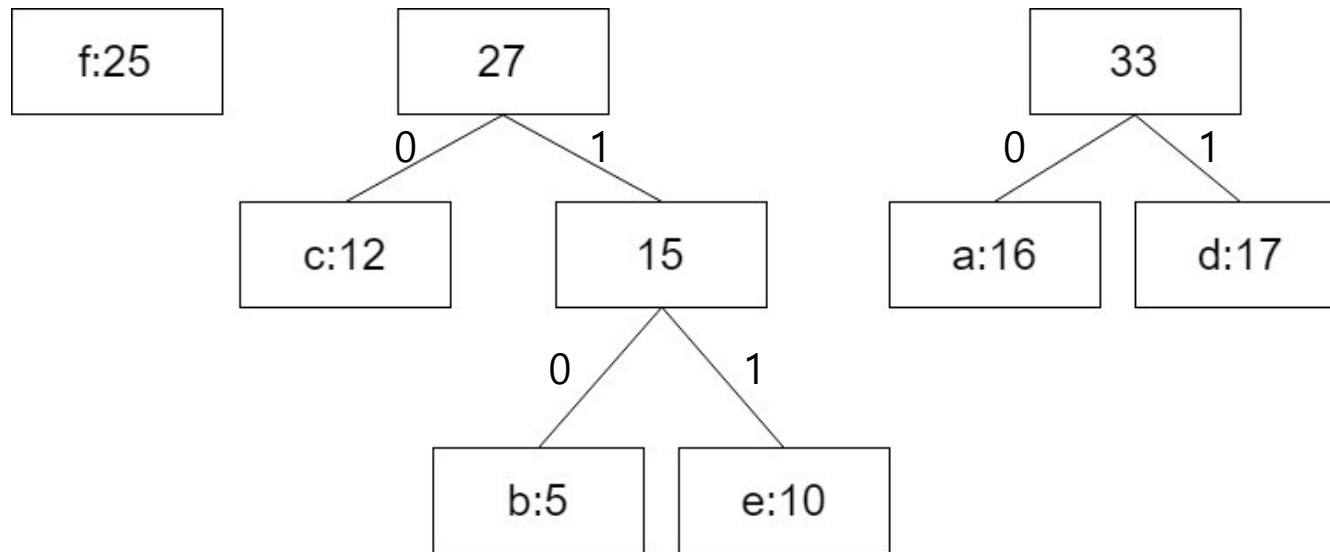
Huffman Code



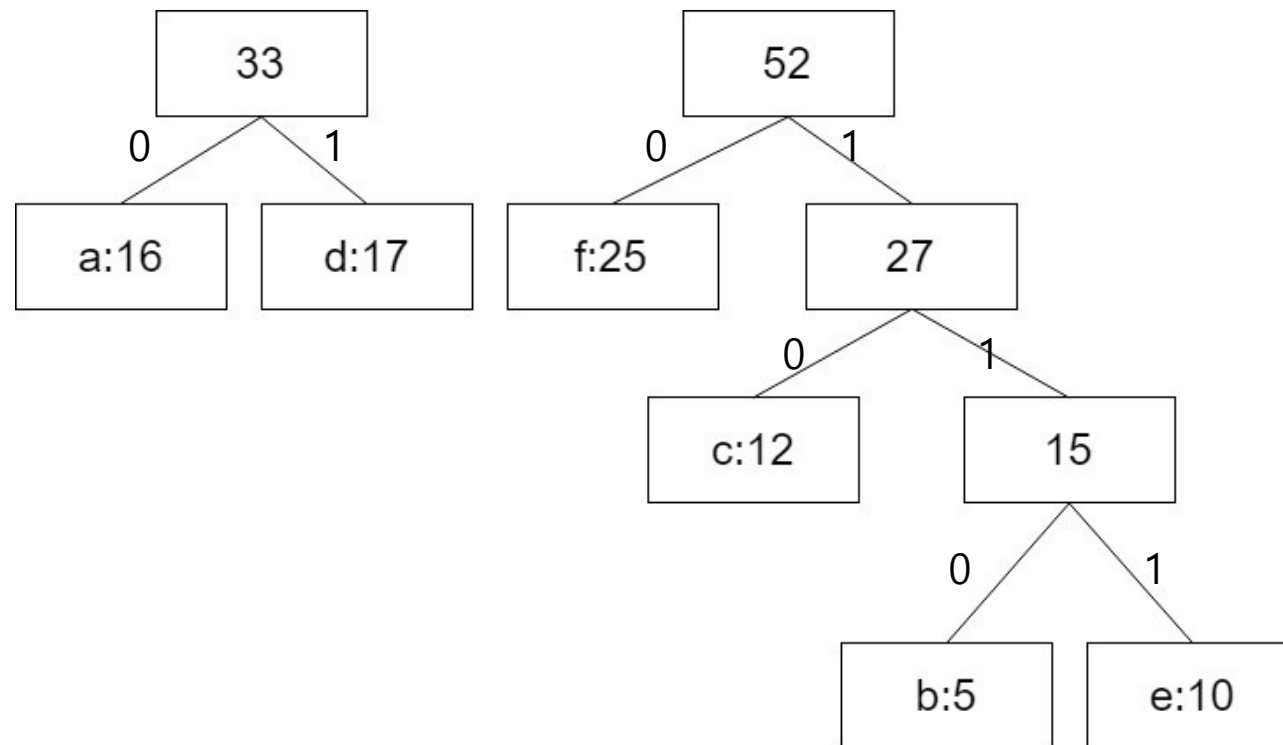
Huffman Code



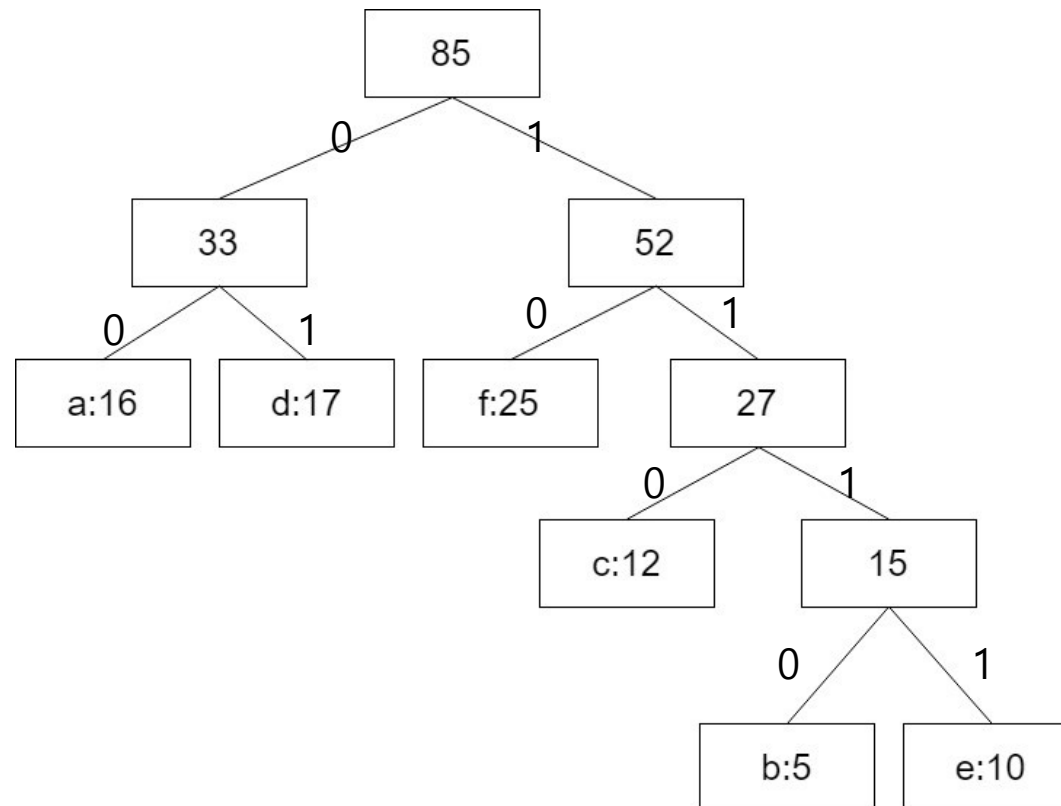
Huffman Code



Huffman Code



Huffman Code

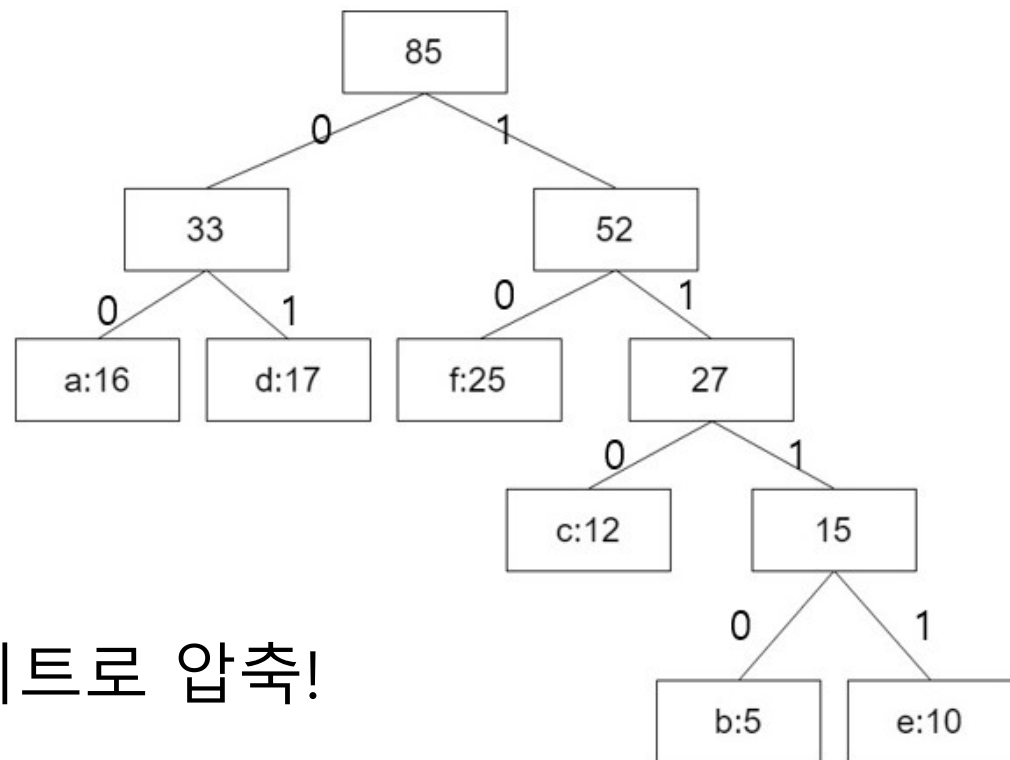


Huffman Code

- a: 00
- b: 1110
- c: 110
- d: 01
- e: 1111
- f: 10

• 문자열을

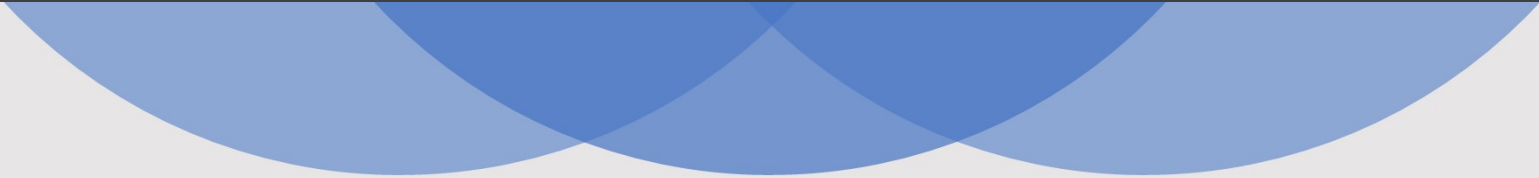
• $15+27+33+52+85 = 212$ 비트로 압축!





관련 문제

11679번 Canvas Painting
10983번 히기카드 (Weather Report)





연습 문제들을 풀어봅시다.