4️⃣

알고리즘 : 그리디 알고리즘

그리디 알고리즘 (Greedy Algorithm)

눈 앞의 이익만 우선 추구하는 알고리즘을 총칭한다.

그리디 알고리즘은 최적화 문제를 대상으로 한다.
대부분 최적해를 보장하지 못한다. 드물게 최적해가 보장되는 경우도 이쌀.

목표 : 최적해를 찾을 수 있으면 찾고, 어려우면 주어진 시간 내에 그런 대로 괜찮은 해를 찾는다.

현재 시점에 가장 이득이 되어 보이는 해를 선택하는 행위를 반복한다.

```
do {
  우선 가장 좋아 보이는 선택을 한다.
} until (해 구성 완료)
```

최소 신장 트리를 찾는 그리디 알고리즘

```
Prim(G, r) {    // 정점 r로부터 시작하여 G=(V, E) 의 최소 신장 트리를 구한다.
    S←Φ; T←Φ;       // S : 정점 집합, T : 간선 집합
    정점 r을 집합 S에 포함시킨다.;
    while (S ≠ V) {
      S에서 V-S를 연결하는 간선들 중 최소 길이인 (x, y) 를 찾음;    //  (x∈S, y∈V-S)
      간선 (x, y)를 T에 포함시킴;
      정점 y를 집합 S에 포함시킴;
    }
}
```

그리디 알고리즘의 전형적인 구조

```
Greedy(C) {   // C : 원소들의 총 집합이다.
    S ←Φ;
    while (C≠Φ and S는 아직 온전한 해가 아니다.) {
      x ← C에서 현재 시점에서 가장 좋아 보이는 원소를 하나 선택한다.;
      집합 C에서 x 제거;     //  C ← C-{x}
      if(S에 x를 더해도 된다.) then S ← S ⋃ {x};
```

```
    }
    if(S가 온전한 해이다.) then return S;
    else return "해 없음!";
}
```

그리디 알고리즘과 최적 해

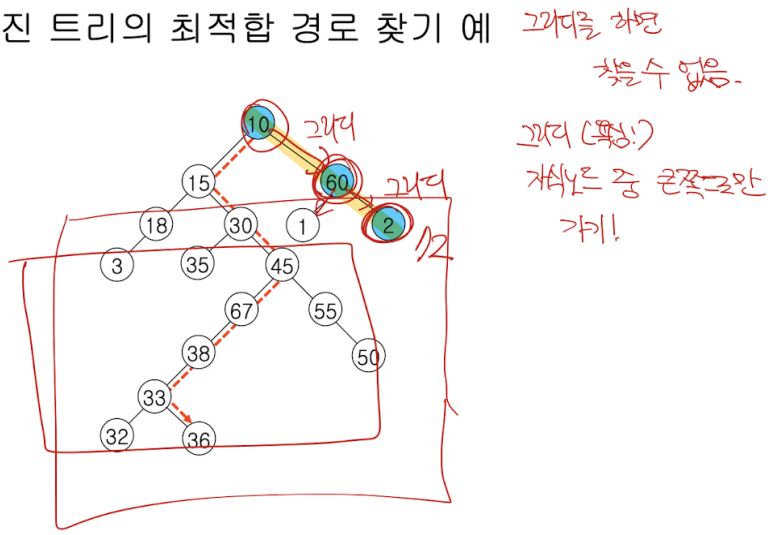
그리디 알고리즘으로 최적해가 보장되지 않는 예

이진 트리의 최적합 경로 찾기

각 노드에 양의 가중치가 할당된 이진트리이다.
각 노드의 가중치는 미리 알 수 없고, 어떤 노드에 이르면 그 노드의 자식 노드 가중치를 알 수 있다. 루트부터 시작해 왼쪽/오른쪽 중 분기할 방향을 매 단계마다 결정한다. 각 경로의 점수는 루트 노드에서 리프 노드에 이를 때까지 만난 노드의 가중치의 합이다.

문제 : 경로의 점수를 최대화하는 경로를 찾아라.

- 이진 트리의 최적합 경로 찾기 예

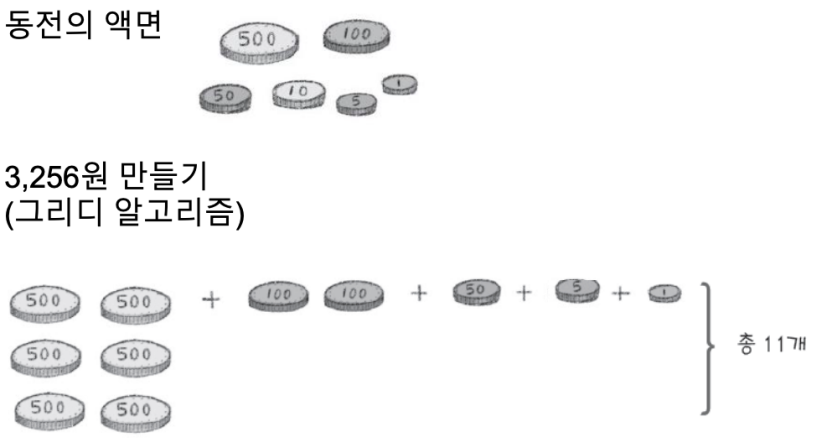


동전 바꾸기

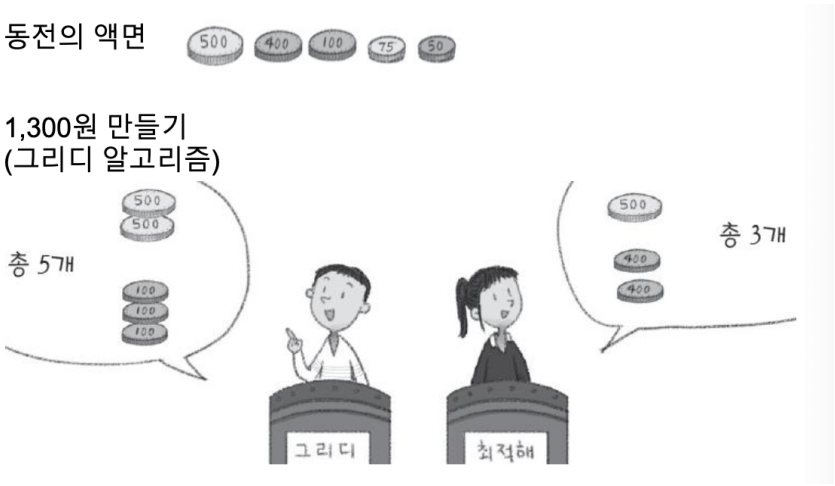
문제 : 동전을 모아 특정 액수를 만든다. 동전의 개수를 최소로 하라.

- 그리디 알고리즘으로 최적해가 보장되는 예

모든 동전의 액면이 바로 아래 액면의 배수가 되면 그리디 알고리즘으로 최적해가 보장된다.
• 예) 500원, 100원, 50원, 10원, 5원, 1원



- 그리디 알고리즘으로 최적해가 보장되지 않는 예
- 예) 500원, 400원, 100원, 75원, 50원



보따리 문제 (0/1 Knapsack Problem)

문제 : 용량이 정해진 보따리에 가치와 부피가 제각각인 물건들을 넣는다. 보따리에 넣은 물건의 총 가치가 최대가 되도록 하라.

- 그리디 알고리즘으로 최적해가 보장되는 예

물건을 자를 수 있다면(Fractional Knapsack Problem) 이 방식으로 최적해를 보장할 수 있다.

- 그리디 알고리즘으로 최적해가 보장되지 않는 예

보따리 용량을 초과하지 않는 한, 단위 부피 당 가치가 큰 물건부터 보따리에 넣는다. 이 그리디 알고리즘은 최적해를 보장하지 못한다.

- 보따리 용량 21L
- 물건의 가격과 부피 — 단위 부피 당 가치
  - A: 250원, 10L — 250/10 = 25원/L
  - B: 450원, 15L — 450/15 = 30원/L
  - C: 270원, 10L — 270/10 = 27원/L
- 그리디 알고리즘 가치가 젤 높네요.
  - 보따리에 B를 넣음 → 물건의 총 가치 450원 B, 15L, 450원
- 최적 해
  - 보따리에 A, C 를 넣음 → 물건의 총 가치 520원

  A+C , 20L , 520원

그리디 알고리즘으로 최적해가 보장되는 예

최소 신장 트리 : Prim(프림) 알고리즘, Kruskal(크루스칼) 알고리즘

```
Prim(G, r) {    // 정점 r로부터 시작하여 G=(V, E) 의 최소 신장 트리를 구한다.
  S←Φ; T←Φ;       // S : 정점 집합, T : 간선 집합
  정점 r을 집합 S에 포함시킨다.;
  while (S ≠ V) {
      S에서 V-S를 연결하는 간선들 중 최소 길이인 (x, y) 를 찾음;   //  (x∈S, y∈V-S)
      간선 (x, y)를 T에 포함시킴;
      정점 y를 집합 S에 포함시킴;
  }
}
```

=> `S에서 V-S를 연결하는 간선들 중 최소 길이인 (x, y) 를 찾음;` 이 부분이 Greedy한 부분이다.

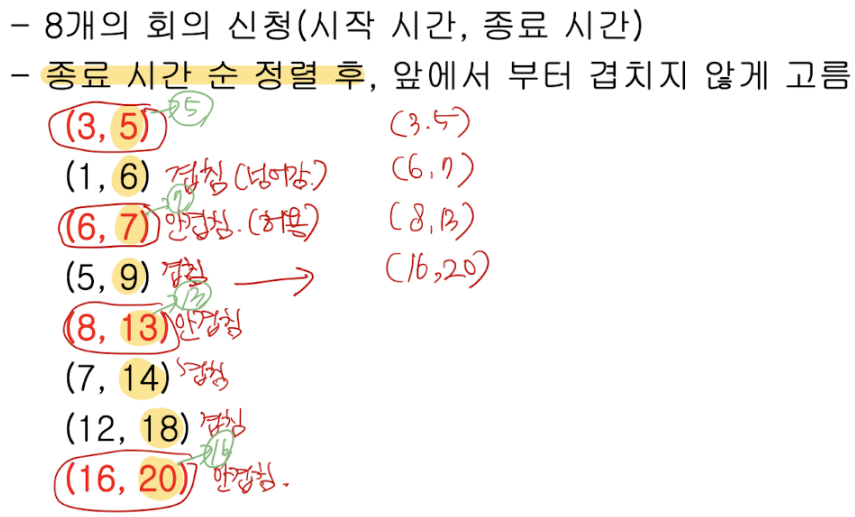
회의실 배정 문제

회사에 회의실이 1개이다. 여러 부서에서 회의실 사용을 신청한다. 회의 시작 시간과 종료 시간을 명시해서 신청한다.

문제 : 겹치는 시간이 없게 가장 많은 수의 회의를 소화하도록 회의실 사용 스케줄을 정하라.

Greedy한 아이디어들

1. 소요 시간이 가장 짧은 회의순 배정
2. 시작 시간이 가장 이른 회의순 배정
3. 종료 시간이 가장 이른 회의순 배정 (-> 이것만 최적해를 보장한다.)



최단 경로 : Dijkstra(다익스트라) 알고리즘

허프만 코딩 알고리즘

요약

1. 그리디 알고리즘은 눈 앞의 이익만 추구하는 알고리즘을 총칭한다.
2. 그리디 알고리즘은 대부분 최적해를 보장하지 못한다.
3. 어떤 문제들을 그리디 알고리즘으로 최적해가 보장된다.
4. 그리디 알고리즘으로 최적해가 보장되지 않는 예는 이진트리의 최적합 경로 찾기, 동전 바꾸기, 보따리 문제가 있다. (동전과 보따리 문제는 조건을 추가하면 최적해를 구할 수 있다.)
5. 그리디 알고리즘으로 최적해가 보장 되는 예는 최소신장트리 (프림알고리즘, 크루스칼 알고리즘), 최단 경로(다익스트라 알고리즘), 회의실 배정 문제, 허프만 코딩 알고리즘이 있다.