

# 브루트 포스 - 재귀 (참고)

최백준 [choi@startlink.io](mailto:choi@startlink.io)

---

# 퇴사

<https://www.acmicpc.net/problem/14501>

- $N+1$ 일이 되는 날 퇴사를 하려고 한다 ( $1 \leq N \leq 15$ )
- 남은  $N$ 일 동안 최대한 많은 상담을 하려고 한다
- 하루에 하나의 상담을 할 수 있고
- $i$ 일에 상담을 하면,  $T[i]$ 일이 걸리고  $P[i]$ 원을 번다

# 퇴사

<https://www.acmicpc.net/problem/14501>

- go(day, sum)
  - day일이 되었다. day일에 있는 상담을 할지 말지 결정해야 한다.
  - 지금까지 얻은 수익은 sum이다

# 퇴사

<https://www.acmicpc.net/problem/14501>

- go(day, sum)
  - day일이 되었다. day일에 있는 상담을 할지 말지 결정해야 한다.
  - 지금까지 얻은 수익은 sum이다
- 정답을 찾은 경우
  - day == n
- 불가능한 경우
  - day > n
- 다음 경우
  - 상담을 한다: go(day+t[day], sum+p[day])
  - 상담을 하지 않는다: go(day+1, sum)

# 퇴사

<https://www.acmicpc.net/problem/14501>

• 소스: <http://codeplus.codes/7e76d2bb354244cb80d926e36eb6ae85>

```
• void go(int day, int sum) {  
    if (day == n+1) {  
        if (ans < sum) ans = sum;  
        return;  
    }  
    if (day > n+1) {  
        return;  
    }  
    go(day+1, sum);  
    go(day+t[day], sum+p[day]);  
}
```

# 퇴사

<https://www.acmicpc.net/problem/14501>

- 앞의 코드를 return값을 이용하게 수정하면 다음과 같이 수정할 수 있다.

```
int go(int day) {
    if (day == n+1) {
        return 0;
    }
```

```
    if (day > n+1) {
        return -inf;
    }
```

```
    int t1 = go(day+1); // x
```

```
    int t2 = go(day+t[day]) + p[day]; // o
```

```
    return max(t1, t2);
```

```
}
```

day 할 복타    얻을 수 있는  
최대 수익

# 퇴사

2/21

최대 수익

7

<https://www.acmicpc.net/problem/14501>

- N = 6이고, 3일까지 다음과 같은 상황이 가능했다고 가정하자



|                | 1일 | 2일 | 3일 | 4일 | 5일 | 6일 |
|----------------|----|----|----|----|----|----|
| 방법 1<br>수익: 35 | O  | O  | X  |    |    |    |
| 방법 2<br>수익: 20 | X  | O  | X  |    |    |    |
| 방법 3<br>수익: 15 | O  | X  | X  |    |    |    |
| 방법 4<br>수익: 55 | X  | O  | O  |    |    |    |

# 퇴사

<https://www.acmicpc.net/problem/14501>

- 4일이 된 경우에 전체 정답은 (1~3일까지 상담한 수익) + (4~6일까지 상담한 수익)으로 계산할 수 있다.

|                | 1일 | 2일 | 3일 | 4일 | 5일 | 6일 |
|----------------|----|----|----|----|----|----|
| 방법 1<br>수익: 35 | 0  | 0  | X  |    |    |    |
| 방법 2<br>수익: 20 | X  | 0  | X  |    |    |    |
| 방법 3<br>수익: 15 | 0  | X  | X  |    |    |    |
| 방법 4<br>수익: 55 | X  | 0  | 0  |    |    |    |



# 퇴사

<https://www.acmicpc.net/problem/14501>

- 4일이 된 경우에 선택할 수 있는 상담의 방법은 1~3일에 영향을 받지 않는다.
- 4일부터 상담을 시작했을 때 4~6일동안 얻을 수 있는 최대 수익은
- 1~3일까지 적절히 상담을 진행하고 4일이 된 후에 4~6일동안 얻을 수 있는 최대 수익과 같다.

|                | 1일 | 2일 | 3일 | 4일 | 5일 | 6일 |
|----------------|----|----|----|----|----|----|
| 방법 1<br>수익: 35 | 0  | 0  | X  |    |    |    |
| 방법 2<br>수익: 20 | X  | 0  | X  |    |    |    |
| 방법 3<br>수익: 15 | 0  | X  | X  |    |    |    |
| 방법 4<br>수익: 55 | X  | 0  | 0  |    |    |    |

방법 4

# 퇴사

<https://www.acmicpc.net/problem/14501>

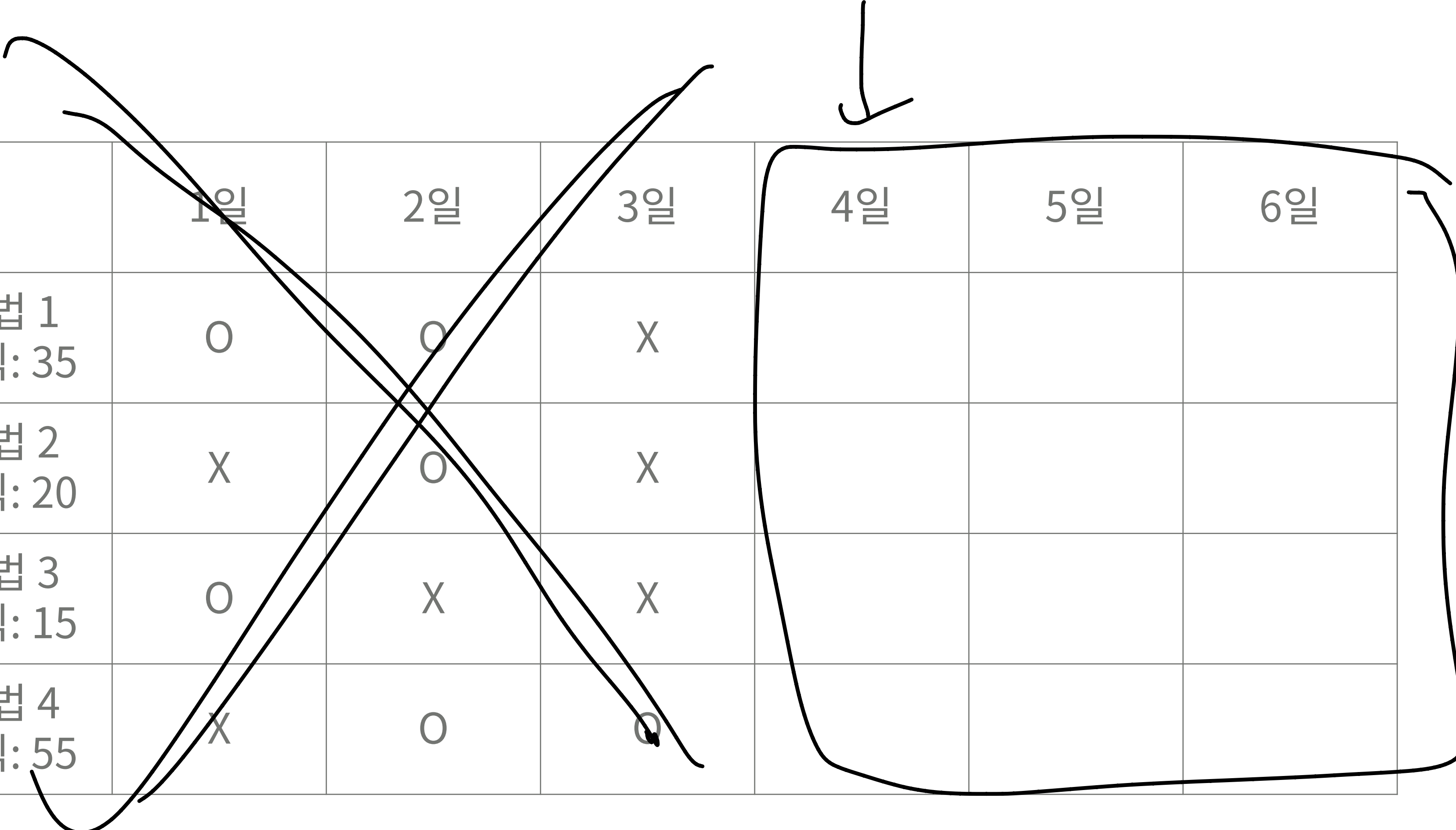
- 따라서, 각각의 경우에 최댓값만 알고 있으면 된다.
- 4일부터 진행해서 얻을 수 있는 최대 수익이 아래의 방법 1, 2, 3, 4 모두 같기 때문이다.

|                | 1일 | 2일 | 3일 | 4일 | 5일 | 6일 |
|----------------|----|----|----|----|----|----|
| 방법 1<br>수익: 35 | O  | O  | X  |    |    |    |
| 방법 2<br>수익: 20 | X  | O  | X  |    |    |    |
| 방법 3<br>수익: 15 | O  | X  | X  |    |    |    |
| 방법 4<br>수익: 55 | X  | O  | O  |    |    |    |

# 퇴사

<https://www.acmicpc.net/problem/14501>

- 이를 바탕으로  $i$ 일부터 상담을 진행했을 때 얻을 수 있는 최대 수익을 저장해 메모이제이션을 구현할 수 있다.



|                | 1일 | 2일 | 3일 | 4일 | 5일 | 6일 |
|----------------|----|----|----|----|----|----|
| 방법 1<br>수익: 35 | 0  | 0  | X  |    |    |    |
| 방법 2<br>수익: 20 | X  | 0  | X  |    |    |    |
| 방법 3<br>수익: 15 | 0  | X  | X  |    |    |    |
| 방법 4<br>수익: 55 | X  | 0  | 0  |    |    |    |

# 퇴사

<https://www.acmicpc.net/problem/14501>

```
int go(int day) {  
    if (day == n+1) return 0;  
    if (day > n+1) return -inf;  
    if (d[day] != -1) {  
        return d[day];  
    }  
    int t1 = go(day+1); // x  
    int t2 = p[day] + go(day+t[day]); // o  
    d[day] = max(t1, t2);  
    return d[day];  
}
```

$d[day] = \text{go}(day)$  이  
2(핀)  
  
= day 이 부터  
오른 쪽으로  
3(44)

# 퇴사

<https://www.acmicpc.net/problem/14501>

- 소스: <http://codeplus.codes/3492fb7d88cd43d49362e7d9f16ee87d>

