스피드러너 스트리머의 궁극의 도전 문제

문제

유명한 게임 스트리머인 '갓게이머'는 하드코어 게임 '갓슬레이어'의 스피드러닝 대회에 참여하게 되었다. 경쟁자들을 이길 수 있는 최적화된 플레이를 위해서, 경쟁에 도움이 되는 프로그램을 제작하고자 한다.

'갓슬레이어'는 총 `N`개의 보스로 이루어져 있다. 게이머는 보스를 잡는 순서를 마음대로 결정할 수 있으며, 각 보스를 물리치는 데에는 기본적으로 특정한 시간이 소요된다. 보스를 물리치면 특수한 기술을 얻게 되는데, 이 기술은 바로 다음 보스와의 전투에서만 사용할 수 있으며 보스와의 전투 시간을 특정 시간만큼 줄여준다. 이 기술은 바로 다음 보스에게만 사용할 수 있으며, 다른 보스에게는 사용할 수 없다.

보스를 물리치는 데에 소요되는 시간과, 해당 보스를 물리쳤을 때 얻는 기술의 위력은 각각 정수 배열 `boss`, `skill`로 주어진다.

'갓게이머'는 게임을 완료하는 데에 필요한 최소의 시간을 계산하는 프로그램을 제작하고자 한다. 즉, 보스를 물리치는 순서를 달리하여 달성할 수 있는 최소의 게임 시간을 구하시오.

입력설명

- `0 < N <= 10`
- `0 < boss[i] <= 100`
- `0 < skill[i] <= 100`

매개변수 형식

- `N = 5`
- `boss = [10, 8, 12, 5, 20]`
- `skill = [0, 19, 5, 10, 3]`

반환값 형식

`18`

예시 입출력 설명

예시1

- 입력
  - `N = 5`
  - `boss = [10, 8, 12, 5, 20]`
  - `skill = [0, 19, 5, 10, 3]`
- 반환값
  - `18`

- 설명

  아래 순서로 보스를 물리쳤을 때 최선의 결과이다.
  - `1`번 보스를 시간 `8`를 들여서 제거. `19`스킬 획득
  - `4`번 보스를 시간 `20-19=1`을 들여서 제거. `3`스킬 획득
  - `3`번 보스를 시간 `5-3=2`를 들여서 제거. `10`스킬 획득
  - `2`번 보스를 시간 `12-10=2`를 들여서 제거. `5`스킬 획득
  - `0`번 보스를 시간 `10-5=5`를 들여서 제거. 모든 보스 제거 완료
  - 총 `8+1+2+2+5=18`의 시간이 소요되었다.

예시2

- 입력
  - `N = 7`
    0 1 2 3 4 5 6
  - `boss = [75, 25, 70, 46, 60, 7, 85]`
  - `skill = [65, 28, 78, 29, 37, 63, 89]`
- 반환값
  - `30`

- 설명

아래 순서로 보스를 물리쳤을 때 최선의 결과이다.
- `5`번 보스를 시간 `7`을 들여서 제거. `63`스킬 획득
- `2`번 보스를 시간 `70-63=7`을 들여서 제거. `78`스킬 획득
- `6`번 보스를 시간 `85-78=7`을 들여서 제거. `89`스킬 획득
- `0`번 보스를 시간 `75-89->0`을 들여서 제거. `65`스킬 획득
- `4`번 보스를 시간 `60-65->0`을 들여서 제거. `37`스킬 획득
- `3`번 보스를 시간 `46-37=9`를 들여서 제거. `29`스킬 획득
- `1`번 보스를 시간 `25-29->0`을 들여서 제거. 모든 보스 제거 완료
- 총 `7+7+7+9=30`의 시간이 소요되었다.

제로 카페의 비결 해설

- **Python3 정답 코드 예시**

```
def solution(N, boss, skill):
    dp = dict()
    result = float('inf')

    def dfs(boss_status, current_skill, time_spent):
        nonlocal result

        status = boss_status + current_skill * 2**N

        if status in dp:
            time_spent = min(dp[status], time_spent)
```

```
        dp[status] = time_spent

        if time_spent >= result:
            return

        if boss_status == 2**N - 1:
            result = min(result, time_spent)
            return

        for i in range(N):
            if boss_status & (1 << i) == 0:
                dfs(boss_status | (1 << i),
                    skill[i],
                    time_spent + max(0, boss[i] - current_skill))

    initial_boss_status = 0
    dfs(initial_boss_status, 0, 0)
    return result
```

- **Java 정답 코드 예시**

```
import java.util.HashMap;
import java.util.Map;

class Solution {
    Map<Integer, Integer> dp = new HashMap<>();
    int result = Integer.MAX_VALUE;
    int N;
    int[] boss;
    int[] skill;
    int twoPowN;

    public int solution(int N, int[] boss, int[] skill) {
        this.N = N;
        this.boss = boss;
        this.skill = skill;
```

```
        twoPowN = 1 << N;
        int initialBossStatus = 0;
        dfs(initialBossStatus, 0, 0);
        return result;
    }

    void dfs(int bossStatus, int currentSkill, int timeSpent) {
        int status = bossStatus + currentSkill * twoPowN;

        if (dp.containsKey(status)) {
            timeSpent = Math.min(dp.get(status), timeSpent);
        }
        dp.put(status, timeSpent);

        if (timeSpent >= result) {
            return;
        }

        if (bossStatus == twoPowN - 1) {
            result = Math.min(result, timeSpent);
            return;
        }

        for (int i = 0; i < N; i++) {
            if ((bossStatus & (1 << i)) == 0) {
                dfs(bossStatus | (1 << i),
                    skill[i],
                    timeSpent + Math.max(0, boss[i] - currentSkill));
            }
        }
    }
}
```