2021 여름학기 동국대학교 SW역량강화캠프

17일차. 다이나믹 프로그래밍 2

● 다이나믹 프로그래밍(DP)

▶ 값의 재활용 (중복되는 계산은 한 번만)

▶ 큰 문제를 작은 문제들로 나누어 해결하는 방법

▶ 동일한 형태의 문제들끼리의 관계식을 찾는 것이 핵심

● 용돈 받는 윌리 (4094)

문제

용돈을 다쓴 윌리는 돈을 벌기 위해 N일동안 부모님의 가게를 도우려고 한다.

윌리가 일하는 가게에서는 설거지하기, 요리하기, 마당쓸기 세 가지의 일이 있다.

윌리는 지루한 일을 굉장히 싫어하기 때문에, 이틀연속 같은 일을 하는 것은 불가능하다.

i일에 설거지하기, 요리하기, 마당쓸기에 대한 용돈이 주어질 때, 벌 수 있는 돈의 최대를 구하자.

입력

첫째 줄에 윌리가 일하는 날 N이 주어진다. ($1 \leq N \leq 10^5$)

둘째 줄부터 N개의 줄에 설거지, 요리, 마당쓸기를 할때의 용돈 x_i, y_i, z_i들이 주어진다. ($1 \leq x_i, y_i, z_i \leq 10^4$)

문제 해설

▶ 출력: N일 동안 일해서 받을 수 있는 용돈의 최대값

▶ dp(x) = 첫 x일 동안 일해서 받을 수 있는 용돈의 최대값

▶ dp(x)와 dp(x−1)의 관계식은?

▶ x일자에 어느 것을 선택했는가에 따라 x−1일에 선택할 수 있는 값이 달라짐

▶ dp(x, i) = 첫 x일 동안 일해서 받을 수 있는 용돈의 최대값, x일자에는 작업 i 를 진행

▶ dp(x, i) = max(dp(x−1, j) , dp(x−1, k)) + (x일에 i를 진행해서 받는 돈)

▶ dp(n, *) 의 최대값 출력

10	40	70
20	50	80
30	60	90

10	40	70
max(70, 40) +20 = 90	max(10, 70) +50 = 120	max(10, 40) +80 = 120
max(120, 120) +30 = 150	max(90, 120) +60 = 180	max(90, 120) +90 = 210

핵심 코드

```
for(int i=1; i<=N; i++) {
    st = new StringTokenizer(br.readLine());
    int x = Integer.parseInt(st.nextToken());
    int y = Integer.parseInt(st.nextToken());
    int z = Integer.parseInt(st.nextToken());
    dp[i][0] = Math.max(dp[i-1][1], dp[i-1][2]) + x;
    dp[i][1] = Math.max(dp[i-1][0], dp[i-1][2]) + y;
    dp[i][2] = Math.max(dp[i-1][0], dp[i-1][1]) + z;
}
System.out.println(Math.max(dp[N][0], Math.max(dp[N][1], dp[N][2])));
```

● 증가하는 가장 긴 수열 찾기(4079)

문제

북극곰 윌리는 주어진 수열에서 몇개를 뽑아 증가하는 수열을 만들려고 한다.

수열의 순서는 바꿀 수 없다.

수열 {10, 40, 20, 50, 30} 이 있을 경우, 증가하는 수열의 최대길이는 {10, 20, 30}으로 3이다.

윌리를 도와 증가하는 수열의 최대 길이를 구하자.

입력

첫째 줄에 수열의 길이 N이 주어진다. ($1 \leq N \leq 1000$)

둘째 줄에는 수열을 이루는 값 X_i들이 주어진다. ($1 \leq X_i \leq 1000$)

▶ DP를 이용한 풀이

▶ 출력: 가장 긴 LIS(최장 증가 부분 수열)의 길이

▶ dp(x) = 배열의 [1,x] 에서 나오는 LIS의 길이

▶ 여전히 관계식을 찾기 힘들다.

▶ dp(x) = 맨 마지막 원소가 배열의 x번째 원소인 LIS의 길이

▶ dp(x) = 맨 마지막 원소가 배열의 x번째 원소인 LIS의 길이

▶ dp(i) = 1 + max(dp(j)) (j < i and arr[j] < arr[i])

▶ dp(*) 의 최대값 출력

1	6	2	8	5	7	4	3

1	2	2	3	3	4	3	3

핵심 코드

```
int ans = 0;
for (int i = 1; i <= N; i++) {
    dp[i] = 1;
    for (int j = 1; j < i; j++) {
        if (arr[j] < arr[i]) {
            dp[i] = Math.max(dp[i], dp[j] + 1);
        }
    }
    ans = Math.max(dp[i], ans);
}
System.out.println(ans);
```

● 욕심 많은 윌리 (4100)

문제

헬로바자회에서 N개의 물건을 나눠주는데, 각 물건은 무게 w_i와 가치 v_i가 정해져있다.

욕심이 많은 윌리는 물건을 다 가져가고 싶지만, 가방이 낡아 W보다 무거운 무게를 담으면 가방이 끊어지고 만다.

윌리가 가방이 끊어지지 않는 선에서 가져갈 수 있는 물건 가치의 합의 최대를 구하자.

입력

첫째 줄에 물건의 개수 $N(1 \leq N \leq 100)$과 가방의 한계 $W(1 \leq W \leq 10^5)$가 주어진다.

둘째 줄부터 N개의 줄에 물건의 무게 $x_i(1 \leq x_i \leq W)$와 물건의 가치 $v_i(1 \leq v_i \leq 10^9)$이 주어진다.

▶ DP를 이용한 풀이

▶ 출력: 무게 W 제한 이내에서 고를 수 있는 물건의 최대 가치합

▶ dp(x) = 무게 x로 고를 수 있는 물건의 최대 가치합

▶ 중복해서 고르는 경우 제외 필요 (고르는 순서 상관 X)

▶ dp(x, i) = (1~i)번 물건만 사용하여 무게 x로 고를 수 있는 물건의 최대 가치합

▶ dp(x, i) = (1~i)번 물건만 사용하여 무게 x로 고를 수 있는 물건의 최대 가치합

▶ dp(x, i) = max(dp(x − w[i], i − 1) + v[i] , dp(x, i − 1))

▶ dp(* , N) 의 최대값 출력

	0	1	2	3	4	5	6	7	8	9
(3,6)	0	0	0	6	0	0	0	0	0	0
(4,3)	0	0	0	6	3	0	0	9	0	0
(5,5)	0	0	0	6	3	5	0	9	11	8
(4,7)	0	0	0	6	7	5	0	13	11	12

핵심 코드

```
dp[0][0] = 0L;
for (int i = 1; i <= W; i++)
    dp[0][i] = -1L;

for (int i = 1; i <= N; i++) {
    for (int j = 0; j <= W; j++) {
        dp[i][j] = dp[i - 1][j];
        if (j - w[i] >= 0) {
            dp[i][j] = Math.max(dp[i][j], dp[i - 1][j - w[i]] + v[i]);
        }
    }
}

Long ans = 0L;
for (int i = 0; i <= W; i++) {
    ans = Math.max(dp[N][i], ans);
}
System.out.println(ans);
```