2156_포도주시식

이준

DP 점화식 구하기

- '연속으로 3잔을 마실 수 없다'가 point!!
- 3잔을 연속으로 마시지 않는 경우는

1) (i - 3)번째까지의 dp + (i - 1)번째 + (i)번째
2) (i - 2)번째까지의 dp + (i)번째
3) (i - 1)번째까지의 dp

중 가장 큰 값을 갖는 경우를 dp[i]로 저장

CODE로 보자 (DP)

```
dp[1] = podo[1];
dp[2] = podo[1] + podo[2];
dp[3] = max(podo[1] + podo[3], max(podo[2] + podo[3], podo[1] + podo[2]));

for (int i = 4; i <= n; i++) {
    dp[i] = max(dp[i - 3] + podo[i - 1] + podo[i], max(dp[i - 2] + podo[i], dp[i - 1]));
}
```

1) 경우

2) 경우

3) 경우

CODE로 보자 (전체)

```cpp
#include <iostream>
#include <algorithm>
using namespace std;

int main() {
    int n;
    cin >> n;
    int podo[10002] = {0, };
    int dp[10002] = {0, };

    for (int i = 1; i <= n; i++) {
        cin >> podo[i];
    }

    dp[1] = podo[1];
    dp[2] = podo[1] + podo[2];
    dp[3] = max(podo[1] + podo[3], max(podo[2] + podo[3], podo[1] + podo[2]));

    for (int i = 4; i <= n; i++) {
        dp[i] = max(dp[i - 3] + podo[i - 1] + podo[i], max(dp[i - 2] + podo[i], dp[i - 1]));
    }

    if (n >= 4) cout << max(dp[n - 2], max(dp[n - 1], dp[n])) << endl;
    else cout << dp[n] << endl;


    return 0;
}

```

시간복잡도

```cpp
#include <iostream>
#include <algorithm>
using namespace std;

int main() {
    int n;
    cin >> n;
    int podo[10002] = {0, };
    int dp[10002] = {0, };

    for (int i = 1; i <= n; i++) {
        cin >> podo[i];
    }

    dp[1] = podo[1];
    dp[2] = podo[1] + podo[2];
    dp[3] = max(podo[1] + podo[3], max(podo[2] + podo[3], podo[1] + podo[2]));

    for (int i = 4; i <= n; i++) {
        dp[i] = max(dp[i - 3] + podo[i - 1] + podo[i], max(dp[i - 2] + podo[i], dp[i - 1]));
    }

    if (n >= 4) cout << max(dp[n - 2], max(dp[n - 1], dp[n])) << endl;
    else cout << dp[n] << endl;


    return 0;
}
```

O(N)

O(N)

END!! 후하