Q. Consider a row of n coins of values $v_1, v_2, \ldots, v_n$ where n is even. We play a game against an opponent by alternating turns. In each turn, a player selects either the first or the last coin from the row, removes it from the row permanently, and receives the value of the coin.

Determine the maximum possible amount of money we can definitely win if we move first. [**Sanjib Chowdhury**]

$\Rightarrow$ State: $DP(i,j)$ $i \Rightarrow$ index of first remaining coin, $j \Rightarrow$ index of last remaining coin

State value function.

$V(i,j) \Rightarrow$ maximum amount of money the current player can guarantee from remaining coins with values $v_i, v_{i+1}, \ldots, v_j$.

Initialization:

$$\begin{aligned} V(i,j) &= 0 \quad , \text{ if } i > j \\ &= v(i) \quad , \text{ if } i = j \end{aligned}$$

Action:

① Select the first coin $a_1 = i$ (chosen by us)

Then value $= v_i$ and remaining $[i+1, j]$

Then opponent can choose either $i+1$ or j

$\hookrightarrow$ if $(i+1)$ chosen, then leaving $[i+2, j]$

$\hookrightarrow$ if j chosen, then leaving $[i+1, j-1]$

Since opponent wants to minimize our earnings, our guaranteed future earnings are:

$$\min\Big\{V(i+2,j),\ V(i+1,j-1)\Big\}$$

So, the action value for action a_1

$$Q(i,j,a_1) = v_i + \min\Big\{V(i+2,j),\ V(i+1,j-1)\Big\}$$

② Select the last coin, $a_2 = j$

value $= v_j$

Similarly,

$$Q(i,j,a_2) = v_j + \min\Big\{V(i+1,j-1),\ V(i,j-2)\Big\}$$

Bellman equation.

$$V(i,j) = \max_{a \in A(i,j)} Q(i,j,a)$$

overall,

$$V(i,j) = \max\begin{Bmatrix} v_i + \min\big\{V(i+2,j),\ V(i+1,j-1)\big\}, \\ v_j + \min\big\{V(i+1,j-1),\ V(i,j-2)\big\} \end{Bmatrix}$$

Space complexity:- $1 \le i \le j \le n$

max. no. of states $= O(n^2)$

Time Complexity:- $O(n^2) \times O(1)$

$= O(n^2)$

Algorithm 1 Coin Game

Require: Coin values $v_1, \ldots, v_n$ (n even)
Ensure: Maximum amount guaranteed for the first player

```
V(i, j) ← 0 for all i > j
for i = 1 to n do
    V(i, i) ← v_i
end for
for ℓ = 2 to n do
    for i = 1 to n − ℓ + 1 do
        j ← i + ℓ − 1
        a_1 ← v_i + min{V(i + 2, j), V(i + 1, j − 1)}
        a_2 ← v_j + min{V(i + 1, j − 1), V(i, j − 2)}
        V(i, j) ← max{a_1, a_2}
    end for
end for
return V(1, n)
```

Q. We have set of N objects $1, 2, 3, \ldots, N$ arranged in a given order. We want to group these objects in clusters that contains consecutive objects.

For each subset $i, i+1, i+2, \ldots, i+k$ there is an associated cost $c(i, k)$.

We want to find the grouping that minimizes the sum of the clusters cost. **[Sanjib Chowdhury]**

$\Rightarrow$ State: $DP(i)$

$i \Rightarrow$ no. of objects already clustered $\quad i \in \{0, 1, 2, \ldots, N\}$

State value.

$V(i) =$ minimum cost of clustering objects $1, 2, 3, \ldots, i$

Initially, $V(0) = 0$ (if no objects have been clustered.)

For the remaining states $\quad V(i) = +\infty \quad \forall\, i \in 1, 2, \ldots, N$

Action value. At state i, choose the number of consecutive objects to include in the next cluster.

Let action $a = j \quad \forall\, j \in \{i+1, i+2, \ldots, N\}$

This action creates a cluster with objects $\{i+1, i+2, i+3, \ldots, j\}$

Therefore its cost is: $\quad c(i+1,\, j-i)$

Therefore, $\quad V(j) = V(i) + c(i+1,\, j-i)$

Bellman equation.

$$V(j) = \min_{0 \le i \le j} \Big\{ V(i) + c(i+1,\, j-i) \Big\}$$

Space complexity.

Only one value per state: $O(N)$

Time complexity.

For each j, we examine all partition points i.

$j = 1, 2, 3, \ldots, N$

Total no. of candidates: $1 + 2 + 3 + \cdots + N = O(N^2)$

Algorithm 2 Optimal Clustering of Consecutive Objects

Require: N objects, cost function $c(i, k)$ for the group $i, \ldots, i + k$
Ensure: Minimum total clustering cost

```
V(0) ← 0
for j = 1 to N do
    V(j) ← +∞
    for i = 0 to j − 1 do
        V(j) ← min{V(j), V(i) + c(i + 1, j − i)}
    end for
end for
return V(N)
```

Q. Given, n_1 items of size s_1, n_2 items of size s_2, n_3 items of size s_3.

Each bin has capacity C.

Assume, $s_1, s_2, s_3 \le C$

objective: minimize total number of bins used. **[Sanjib Chowdhury]**

$\Rightarrow$ State. $DP(i, j, k)$

i items of size s_1

j items of size s_2

k items of size s_3

value function.

$V(i, j, k)$

Initially, $V(0, 0, 0) = 0$

Action. At state (i, j, k), choose one feasible bin configuration (a, b, c) such that,

$0 \le a \le i, \quad 0 \le b \le j, \quad 0 \le c \le k$

and $as_1 + bs_2 + cs_3 \le C$

Bellman equation.

$$V(i,j,k) = 1 + \min V(i-a,\ j-b,\ k-c)$$

we add 1 because we have opened one new bin.

Space complexity.

$$O(n_1 \cdot n_2 \cdot n_3)$$

Time complexity.

$$\Big(\text{number of states}\Big)\ *\ \Big(\text{how many actions do we examine per state}\Big)$$

$$= O(n_1 \cdot n_2 \cdot n_3)\ *\ O(n_1 \cdot n_2 \cdot n_3)$$

$$= O\big(n_1^2 \cdot n_2^2 \cdot n_3^2\big)$$

Algorithm 3 Bin Packing with Three Item Sizes

Require: $(n_1, s_1), (n_2, s_2), (n_3, s_3)$, bin capacity C
Ensure: Minimum number of bins

$V(0,0,0) \leftarrow 0$
for $i = 0$ **to** n_1 **do**
    for $j = 0$ **to** n_2 **do**
        for $k = 0$ **to** n_3 **do**
            if $(i,j,k) = (0,0,0)$ **then continue**
            end if
            $V(i,j,k) \leftarrow +\infty$
            for $a = 0$ **to** i, $\quad b = 0$ **to** j, $\quad c = 0$ **to** k **do**
                if $(a,b,c) \neq (0,0,0)$ **and** $as_1 + bs_2 + cs_3 \leq C$ **then**
                $V(i,j,k) \leftarrow \min\{V(i,j,k),\ 1 + V(i-a,\ j-b,\ k-c)\}$
                end if
            end for
        end for
    end for
end for
return $V(n_1, n_2, n_3)$