03 Feb 2025 - Algorithms - Week 06

* Recap

* Proof of : $DFT_n(a_0, \ldots, a_{n-1}) = (b_0, \ldots, b_{n-1})$

$$\Leftrightarrow DFT_{-n}($$

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \omega & \omega & \cdots & \omega \\ \cdots & \vdots & & \cdots \\ \omega^{n-1} & \cdots & & \omega^{n-1} \end{bmatrix} \begin{bmatrix} a_0 \\ \vdots \\ a_{n-1} \end{bmatrix} = \begin{bmatrix} b_0 \\ \vdots \\ b_{n-1} \end{bmatrix}$$

$$\frac{1}{n} DFT_{-n} \, DFT_n \begin{bmatrix} a_0 \\ \vdots \\ a_{n-1} \end{bmatrix} = \frac{1}{n} DFT_{-n} \begin{bmatrix} b_0 \\ \vdots \\ b_{n-1} \end{bmatrix}$$

Show that this is $I_{n \times n}$

i, j^{th} entry $= [1 \quad \omega^{-j} \quad \omega^{-2j} \quad \cdots \quad \omega^{-n}]^T \cdot [\omega^{i-1} \quad \omega^{i-1}$

??

$$1 + \omega + \omega^2 + \cdots + \omega^{n-1} = 0 \qquad \omega = e^{2\pi i/n}$$

$$x^n - 1 = 0$$

$$\omega = e^{2\pi i/15} \qquad k = 2 \; ; \quad n = 15$$

$$\sum_{l=0}^{14} \left(\omega^{2l}\right) = 1 + e^{4\pi i/15} + e^{8\pi i/15} + e^{12\pi i/15} + \underbrace{e^{16\pi i/15}}_{-e^{\pi i/15}} + \cdots$$

$$l = 8 : \quad \omega^{16} = e^{\frac{32\pi i}{16}} = 1$$

$$\therefore \sum_{l=0}^{14} \omega^{2l} = 1 + \omega^2 + \omega^4 + \cdots$$

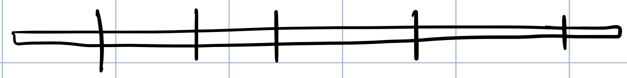
$$+ \underbrace{\omega^{16}}_{1} + \omega^{18} + \cdots$$

??

If k is coprime with n

Dynamic Programming: Rod cutting

→ different prices for different lengths
(not necessarily linear)



How to maximize profit?

length	1	2	3	4	5	6	7	8	9	10
price	1	5	8	9	10	17	17	20	24	30

1.	2.5	2.33	1.25	2 2.9 2.9

Eg. $n = 8$

No cut : 20

1 + 7 : 18

2 + 3 + 3 : 21

optima : 2 + 6 : 22

Greedy approach : pick a length maximizing $\frac{\text{price}}{\text{length}}$

Dynamic Programming :

1. Identify subproblems to be solved
2. Develop a recurrence relation for each subproblem
3. Solve using bottom up approach / tabulate
4. Backtrack

① For $i = 1$ to n

$R(i)$ = max price that can be obtained by breaking and selling a piece of length i

Length	1	2	3	4	5	6	7	8	9	10	i
Price	1	5	8	9	10	17	17	20	24	30	$P(i)$

$$R(1) = 1$$

$$R(2) = \max\left(P(2),\ P(1) + P(1)\right)$$

$$= 5$$

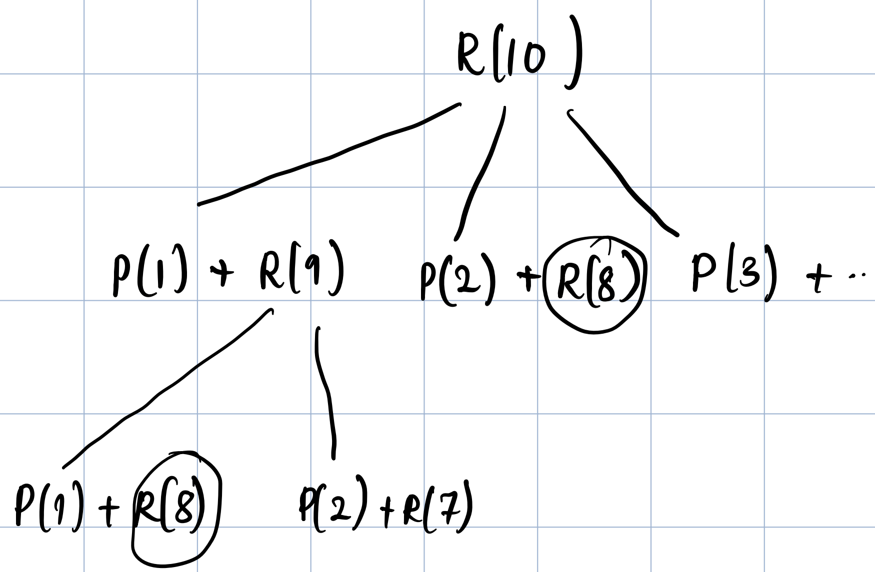
$$\vdots$$

$$R(10) = \max\left(P(10),\ P(1) + R(9),\ P(2) + R(8),\ \ldots\right)$$

1 or 2 or 3 or ... 9

② $R(n) = \max\left(P(n),\ \max\{ P(i) + R(n-i) : 1 \le i \le n-1 \}\right)$

③ Find $R(n)$: return $\max (P(n),\ \max(P(i),$



recursion tree grows exponentially with size

$$R(1) = 1$$

$$R(2) = 5$$

$$R(3) = \max\{8, \; P(1) + R(2)\}$$

$$= 8$$

$$R(4) = \max\{9, \; P(1) + R(3), \; P(2) + R(2), \; P(3) + R(1)\}$$

$$= \max\{9, \; 9, \; 10, \; 8 + 1\}$$

$$= 10$$

$$R(5) = \max\{P(5), \; P(1) + R(4), \; P(2) + R(3), \; P(3) + R(2)\}$$

$$= \max\{10, \; 11, \; 13, \; 13\}$$

$$= 13$$

$$R(6) = \max \{ P(6), P(1) + R(5), P(2) + R(4), P(3) + R(3) \}$$

$$= \max \{ 17, 14, 15, 16 \}$$

$$= 17$$

$$R(7) = \ldots$$

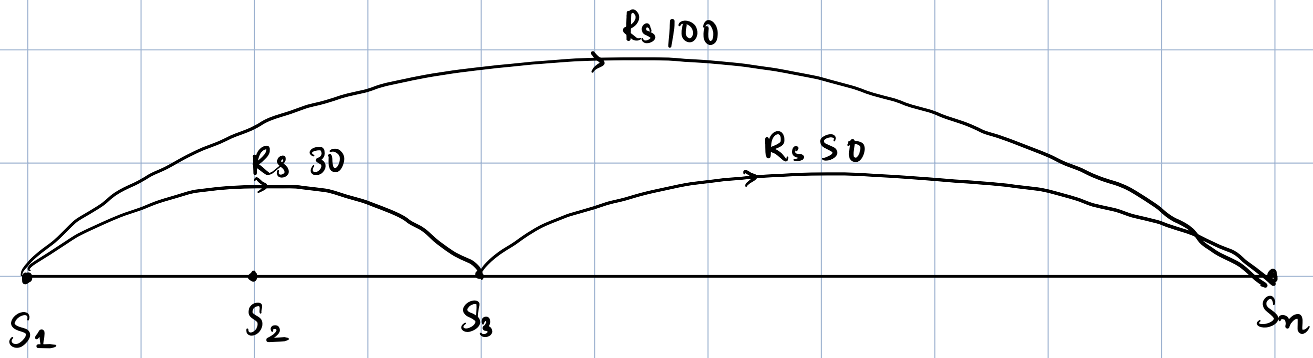
P(i)	1	5	8	9	10	17	17	20	24	30
R(i)	1	5	8	10	13	17	18	22	25	30
J(i)	1	2	3	2	2	6	1	2	3	10

$$R(i) = \max\left(P(i),\ \underbrace{\max_{j<i}\left(P(j) + R(i-j)\right)}_{\text{which } j \text{ gives us max}}\right)$$

Solution corresponding to n: $J(n) = l$

$J(n-l) = l_1$

$J(n-l-l_1) = l_2$



For n stations $S_1, \ldots, S_n$

$i < j$: P_{ij} = ticket price from S_i to S_j

Minimize price of travel S_i to S_j

$$R(i) = \min \left\{ P(i), \min_{j<i} \{ P(i) + R(i-j) \} \right\}$$

$j = 1, \ldots, i-1$

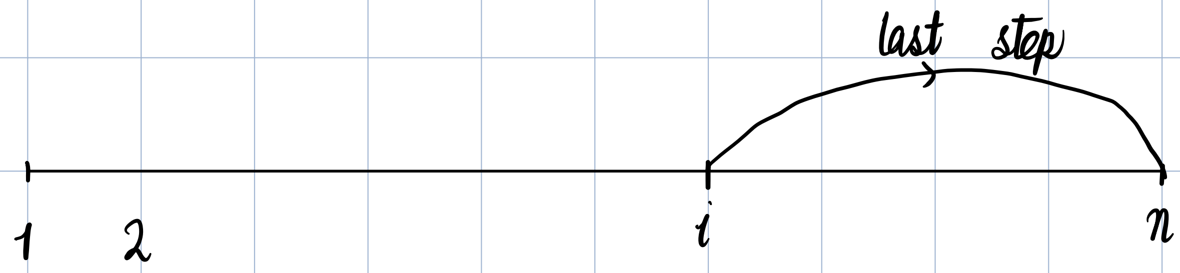
$$R(i,j) = \min\left(P_{ij} \;,\; \min_{k<j} \left(P_{ik} + R(k,j) \right) \right)$$

$k = i+1, \ldots, j$

06 Feb 2025

$R(i)$ = min cost to travel from 1 to i.

Suppose our last step was at i



$$R(n) = \min\left(P_{1,n} \, , \, \min_{i \le n-1} \left(R(i) + P_{i,n} \right) \right)$$

Running time : $O(n^2)$

For $i < j$ $\quad R_{i,j}$ = min cost of travel from i to j.

$$R_{i,j} = \min \left\{ P(i,j),\ \min_{i<k<j} P(i,k) + R(k,j) \right\}$$

1: $R(1,2)$, $R(2,3)$, ... $R(n-1, n)$

2: $R(1,3)$, $R(2,4)$, ...

↙

$\min(P(1,3), R(1,2) + R(2,3))$

How to find subproblems?

I/P : $A[1]$, $A[2]$, ..., $A[n]$

O/P : $f(A[1], \ldots, A[n])$

Subproblems :

→ Find $f(A[1], \ldots, A[i])$

→ Find $f(A[i], \ldots, A[j])$

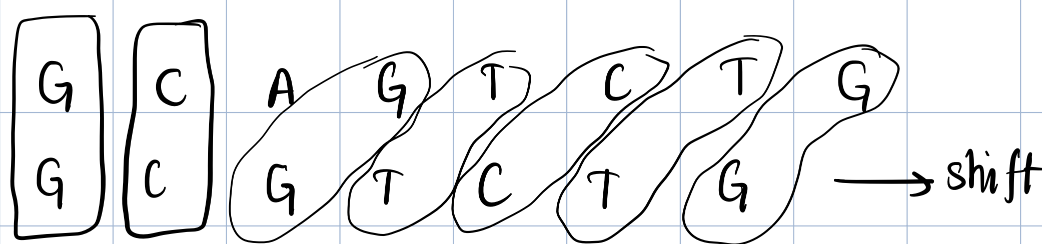
* Similarity in strings (DNA strings)

→ How to define similarity ?

→ String alignment :

not a good

method (shift changes answer)



Spellcheck yay

→ edit distance

Given two strings, how many steps (operations) are needed to get to second string from first.

ED (CLEARS, READS) = ?

① C L E A R S

R L E A R S

② R E A R S

③ R E A D S

ED (ACCTGCAA , CTGCAAG)

A C C T G C A A

(2) C T G C A A

(3) C T G C A A G

ED (ε, GACCT) = 5

$$ED\begin{pmatrix} S_1 & S_2 & \dots & S_n \\ T_1 & T_2 & \dots & T_m \end{pmatrix}$$

$$= ED(S_1 \dots S_{n-1},\ T_1 \dots T_{m-1}) \qquad S_n = T_m$$

$$ED(i, j) = \text{edit distance}\ (S_1 \dots S_i,\ T_1 \dots T_j)$$

If $S_i = T_j$

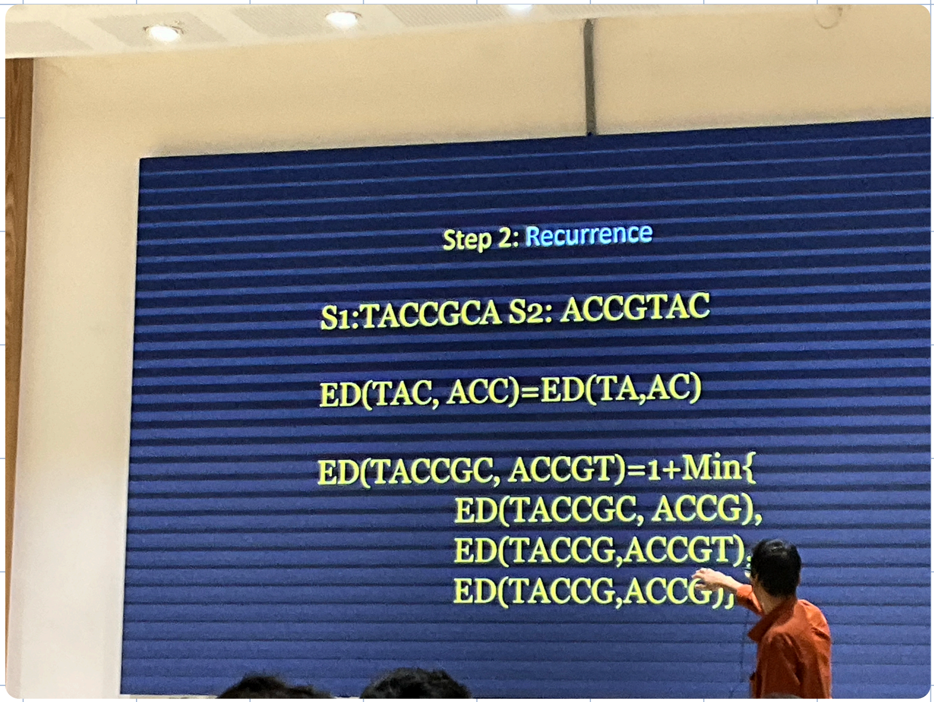
$$ED(i, j) = ED(i-1, j-1)$$

If $S_i \neq T_j$

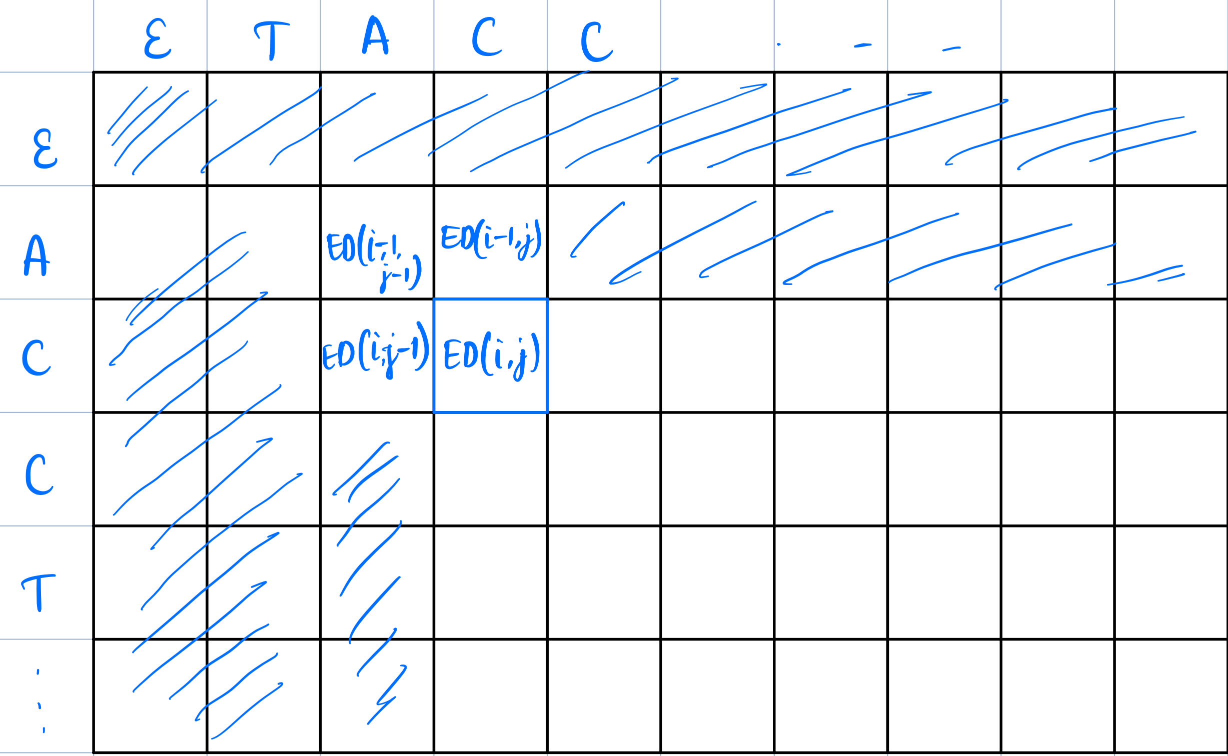
$$\left.\begin{matrix} \underset{\text{Replacement}}{1 + ED(i-1, j-1)} & \underset{\text{Insert}}{1 + ED(i, j-1)} & \underset{\text{Delete}}{1 + ED(i-1, j)} \end{matrix}\right\} \min$$

S_1 : TACCGA

S_2 : ACCG

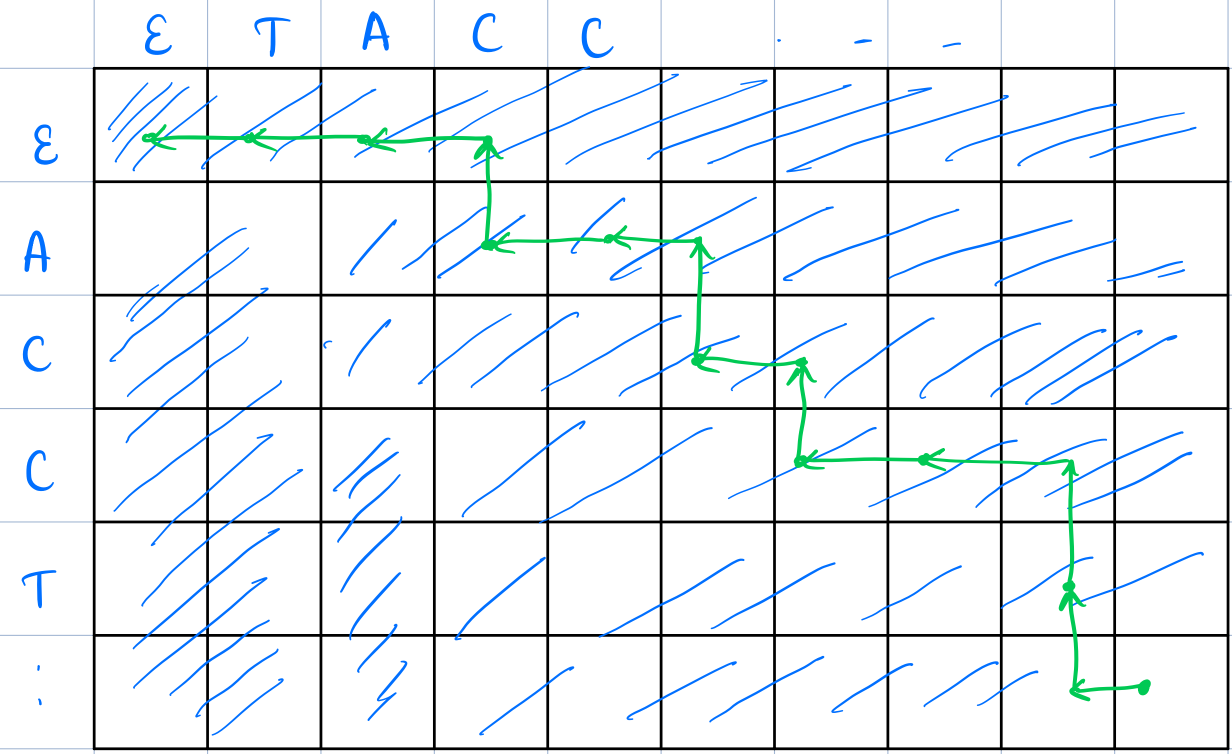


Step 3 : Tabular computation

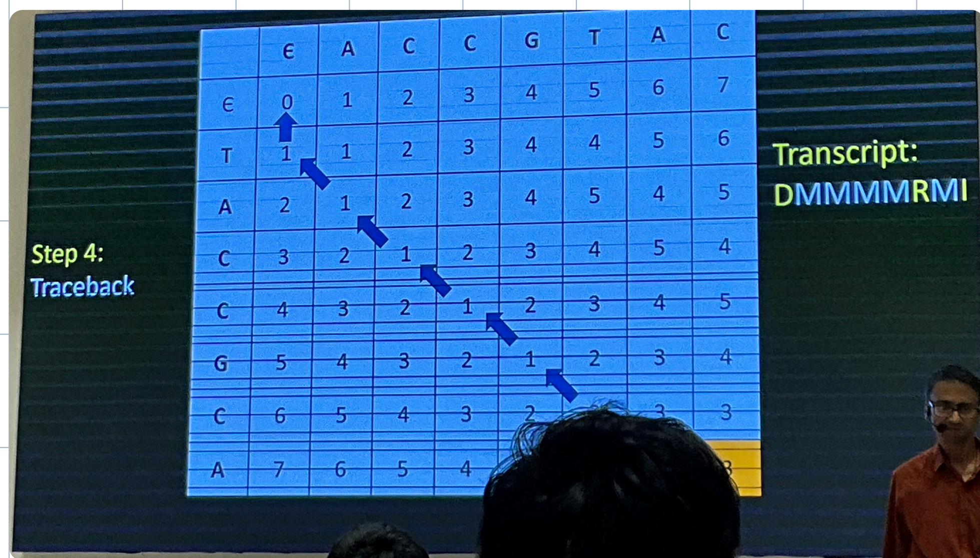


$$ED(i,j) = 1 + \min\left(ED(i-1,j),\ ED(i,j-1),\ ED(i-1,j-1)\right)$$

Step 4: Traceback



ε T A C C
ε
A
C
C
T



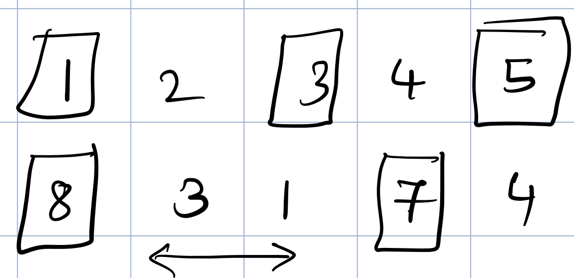
Step 4:
Traceback
Transcript:
DMMMMRMI

Homework

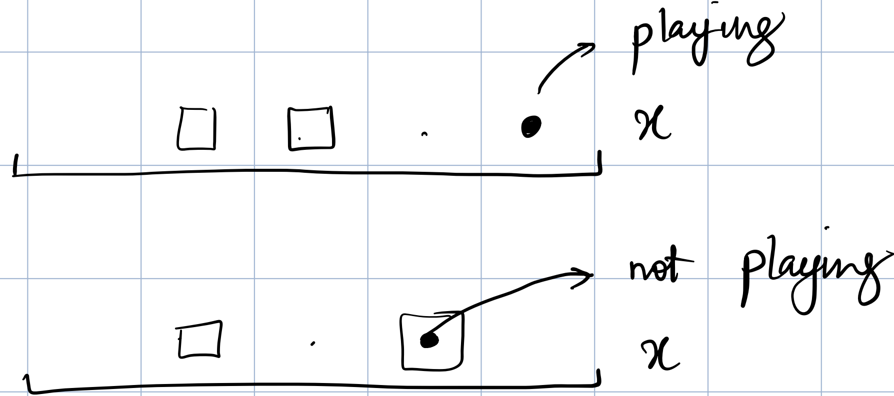
1. Step 1 : Formulation :

(a) Problem we want to solve : maximize the number of activities that I can participate given the constraint that I cannot participate on two consecutive days

(b) Recursion



→ Observation: It would be stupid to not play for 3 or more consecutive days.



max [ ]
↳ stores maximum playables in n days

playing [ ]
↳ whether playing on day x or not

```
if (!playing [n])
    playing [n + 1]
    max [n+1] = max [n] + x
else if ( playing [n - 2] && x > act[n])
    playing [n] = false
    playing [n+1] = true    max [n + 1] = max[n] - act[n]+x
```

else if (playing [n − 3] && act [n − 1] + x > act [n])

playing [n − 1]

!playing [n]

playing [n + 1]

max [n + 1] = max [n] − act [n] + act [n−1] + x

⇝ Convoluted logic  ⟶ bad