N - Stack in An Array

$N \rightarrow$ No of stacks
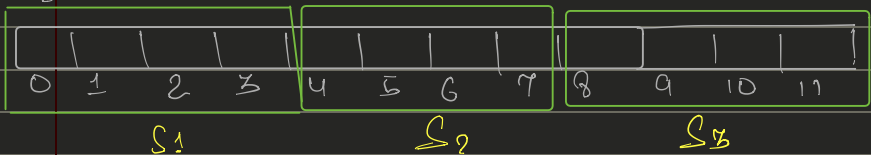
$S \rightarrow$ size of array

$$\boxed{\quad | \quad | \quad | \quad | \quad | \; - - - \; | \quad}$$
$$S-1$$
$$\longleftarrow \qquad S \qquad \longrightarrow$$
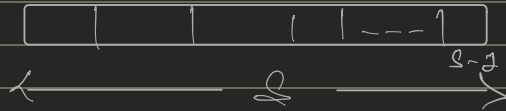
Divide in S/N parts then use them

M-1

ex → $N = 3$, $S = 12$

| 0 | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | 8 | 9 | 10 | 11 |

S_1 S_2 S_3

Problems Not Allocated memory with
Optimization
OR
Not optimal optimization Resource
(Fragmentation)

M-2 → Two additional array →

① → top[] → size → (N) → It stores index of
top element of ith
→ size = S(Main Array) stack.

② → next [] → ⓐ → can Point to next element

after top element.

(6) → It can point to next free space

Initialize-s   N → No of stacks → 3

S → size of array → 6

array →

| | | | | | |
|0|1|2|3|4|5|

Top →

| -1 | -1 | -1 |
| 0 | 1 | 2 |

$\boxed{\text{freespot} = 0}$

next →

| 1 | 2 | 3 | 4 | 5 | -1 |
| 0 | 1 | 2 | 3 | 4 | 5 |

(agla freespot kaha hai)

(1) → Push(x, m)

(1) → // find index
         int index = freespot

(2) → // update freespot

         freespot = next [index]

(3) → // insert in array
         a [index] = x

(4) → // update next
         next [index] = top [m - 1]

(5) → // update top

         top [m - 1] = index

(ii) → POP (m) -> stack

① → find Index

int index = top [m-1]

② → update top

top[m-1] = next [index]

③ → update next

next [index] = freespot

④ → update freespot

freespot = index

⑤ → Return a [index]

DRY - RUN->

$N = 3$, $S = 6$

5	10	6			
0	1	2	3	4	5

Top =

20

-1 ~~0~~ 1	1	-1 ~~-7~~
0	1	2

freespot = ~~0~~ ~~1~~ ~~2~~ 2

next =>

~~1~~ -1	~~2~~ -1	~~3~~ ~~0~~ 4		5		-1
0	1	2	3	4	5

↳ saves top ka
next element

① → Push (5, 1)

int index = 5

$$freespot = next[0] = 1$$

$$a[0] = 5$$
$$next[0] = top[0] = -1$$

$$top[0] = 0$$

② -> Push (10, 2)

-> int index = 1
-> freespot = next[1] = 2

-> $a[1] = 10$
-> $next[1] = top[1]$

-> $top[1] = 1$

③ -> Push (6, 1)

-> index = 2
-> freespot = next[2] = 3

-> $a[2] = 6$

-> $next[2] = top[0] = 0$

-> $top[1] = index = 2$

④ -> Pop(1) ->

-> index = top[0] = 2
-> top[0] = next[2] = 0
-> pop element = arr[2]
-> next[2] = freespot

-> freespot = index