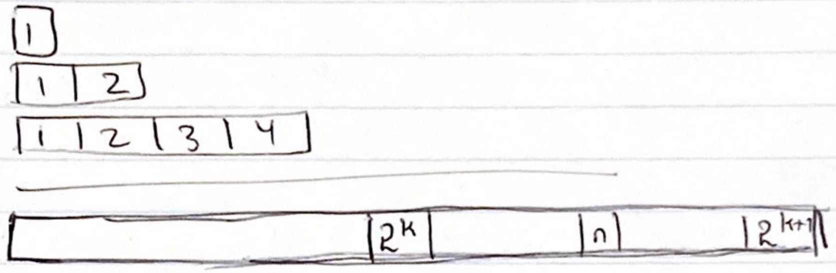
Question 3 :-

Double Approach :- Where you double the existing size of an array.



Consider the aggregate analysis :-

c_i is the cost of ith insertion to the array
if $i = 2^k + 1$, for some $k > 0$, then $c_i = i$,
else $c_i = 1$

$\therefore$ Total cost is given by :-

$$\sum_{i=1}^{n} c_i \leq n + \sum_{j=0}^{\lfloor \log n \rfloor} 2^j$$

$$< n + 2n$$

$$< 3n$$

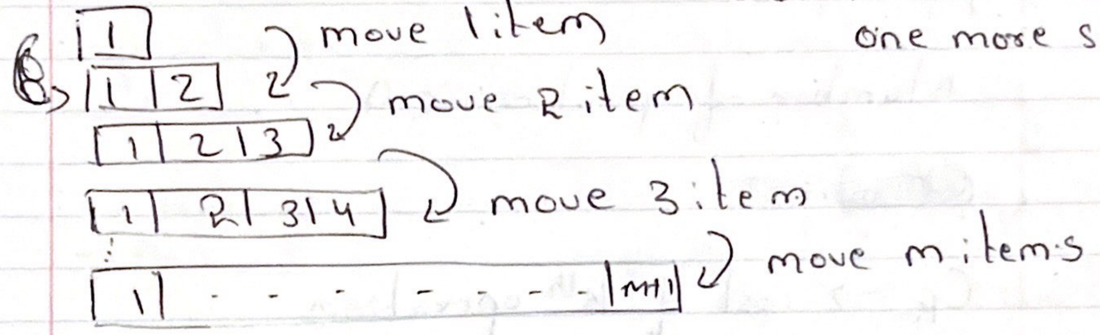
$\therefore$ Cost per insertion is less than $\frac{3\cancel{n}}{\cancel{n}}$

$= 3$

$\therefore$ $O(1)$.

While in case of Increasing Approach :-

(Ik=1 → increase by one more size)



Total cost :- $1+2+\dots+m$

$$= \frac{m(m+1)}{2} \in O(m^2) \rightarrow O(n^2)$$

↳ for n elements

∴ Cost per insertion is $\frac{O(n^2)}{n} = O(n)$

~~Bad~~ Increasing approach is better in case of space complexity as it doesn't assign extra memory.

But in case of Time complexity we can see that Double Approach is better than Increase Approach where Double Approach has time complexity of $O(1)$.

Question 4 :-

Given:-

Number of operations :- n

[illegible]

$C_K \rightarrow$ cost of k^{th} operation

$A_K \rightarrow$ Amortized cost of k^{th} operation.

Let's use [illegible] k credits for each operation whose cost is power of 2 and credit 1 for all other operations

For example.

Operation	Actual cost
1	1
2	2
3	1
4	4
5	1
6	1
7	1
8	8
9	1
10	1

we can see that the actual cost of each operation is either 1 or a power of 2. To determine the amortized cost per operation based on its actual cost. we will use the following charges:-

→ Actual cost is 1 → Let's charge 3units
→ Actual cost is a power of 2
  ↳ let's charge 1unit.

Let calculate the total charge for $n=10$ operation's.

Operations	Actual cost	Amortized cost
1	1	3
2	2	1
3	1	3
4	2	1
5	1	3
6	1	3
7	1	3
8	2	1
9	1	3
10	1	3

∴ ~~Total cost~~
Number of operation with power of $2 = \log_2 n$
Number of operation with actual cost of $1 = n - \log_2 n$

$\therefore$ Total cost $= 3~~\cdot~~(n - \log_2 n) + 1 \times \log_2 n$

$= 3n - 3\log_2 n + \log_2 n$

$= 3n - 2\log_2 n$

Cost per operation

$$= \frac{3n - 2\log_2 n}{n}$$

$$= 3 - \frac{2\log_2 n}{n} \approx \underline{\underline{O(1)}}$$

We can also consider approx Number of of operation ~~≈~~ with power of $2 \approx n/2$ (worst case scenario) also gives Cost per operation $\underline{\underline{O(1)}}$.