ID: 1002256708 Dev P. Patel

HW-17

(1) Inserting n elements using

(a) Aggregate method.

→ The table doubles in size when it runs out of space

→ So, if the Original size is 1, after insertion it doubles to size 2 after 2 more insertions, it doubles to size 4 etc.

→ In general, after k doublings the size 2^k.

* Pseudo Code :-

→ Initialize table with Capacity = 1
for i = 1 to n :
    if table is full:
        → new table
        → Create new table with size 2^k (current size)
→ Copy element from old table to new table.
→ table = new table

→ insert element = i into table
let k : log (n+1) - 1

1002286708

HW-17 Dev P. Patel

Total Cost: $O(n) * k$
$= O(n \log n)$

→ Amortize Cost per insertion = $O(\log n)$

→ Run time pre insertion is $O(\log n)$

→ Total time is $O(n) = \log(n+1)$

(b) Accounting Method:-

* Charge 2 units for each insertion
- When the table doubles in size from m to 2m, Credit m units.
- The Credit exactly pay for the Copy
- Total Credit is $m + 2m + \dots + n/2\, m = O(n)$

* Pseudo Code:-

→ Initialize table with Capacity 1

1002286702

HW-17 Dev P. Patel

for i = 0 to n:
i-th table is full:
new table: create new table with
size 2 * current size.
Copy elements from old table to new table. table = new table.

→ insert element i into table
→ initialize charges = 0
→ initialize credits = 0

→ for i = 1 to n).
charges += 2
if table doubled in size from m to 2m

→ Credits += m.

→ total Charges = 2 * n = O(n)

→ Total Credits = m + 2m + ... + n/2 = O(n).

→ Amortized Cost per insertion.

Total/n = O(n)/n = 1.

→ Runtime per insertion = O(1).

→ Total time = O(n).