Dynamic Array Based Stack : Worst Case Time Analysis

Let’s assume we have N=2^k elements and capacity = 2^k.

1) We perform a push operation and capacity is doubled.

Capacity = 2^(k+1)

Elements= 2^k + 1

Time taken t1= 2^(k+1)+2^k + 1

2) We perform 2^(k-1) pop operations.

Capacity = 2^(k+1)

Elements= 2^(k-1)+1

Time taken t2= 2^(k-1)

3) We perform another pop and capacity is halved.

Capacity = 2^k

Elements= 2^(k-1)

Time taken t3=1+ 2^k + 2^(k-1)

Total operations performed n= 1 + 2^(k-1)+1= 2^(k-1)+2

Total time = t1 + t2 + t3

= 10*n - 18

Amortised Time taken = O((10*n-18)/n)=O(10)=O(1)

NOTE :

1) Cost for creating new array of size n = n

2) Cost for simple push/pop operation = 1 (without changing capacity)

3) Cost of copying n elements into a new array = n