Aditya Dhanaraj Kundu

Design & Analysis of Algorithm

YCS 4002

① Take two numbers ② Add them (take a & b as integer numbers)

```
int main()
{ int a, b ;  ──→ (1)
 scanf("%d %d", &a, &b); →(1)
  a = a+b ; →(1)        total time cycle = 1+1+1 = 3
}
```

① Take the limit of natural numbers ② Calculate the sum.

Take the limit of i as 1 to n for i = 1 to n and sum = sum + i

```
int main(){
  int sum=0, i;  ──→ (1)
  for (i=1; i<11 ; i++)  ──→ 10+1 ; (10)
  { sum = sum+i ; → (10)
  }
}
```

∴ total time cycle = 1+1+(10+1)+10+10
= 2 + 11 + 20 = 33

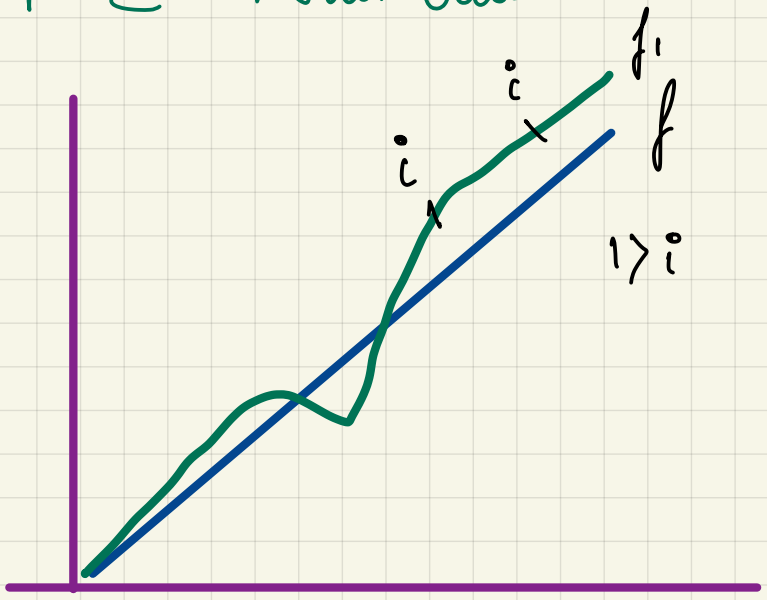
- **Space requirement** :-

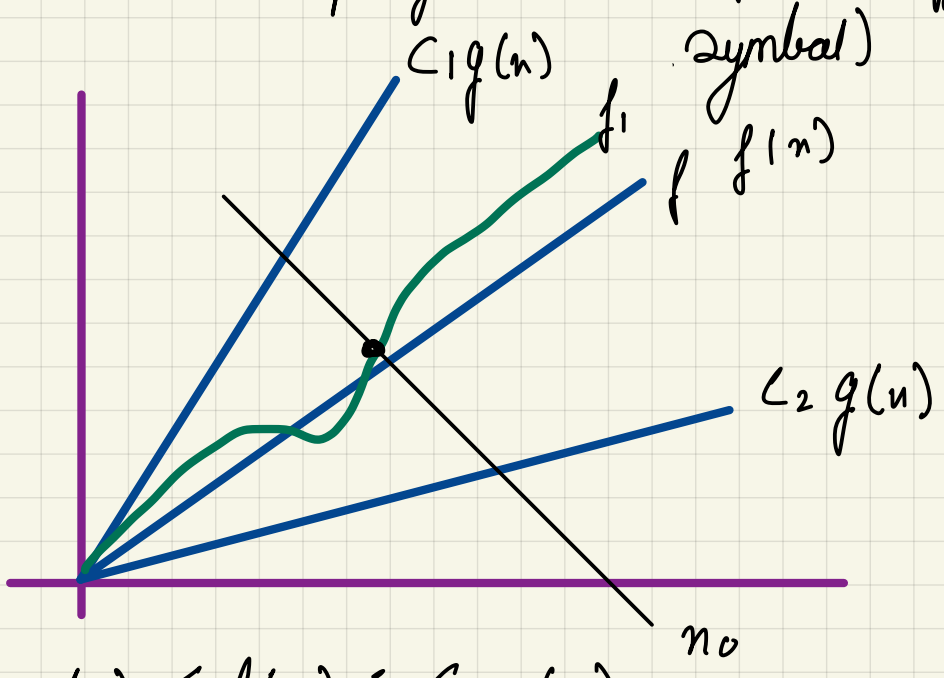
If the no. of variable and the no. of function used is increased then the space requirement will be also increased, so the space complexity is increased in that case.

O, Θ, Ω → asymptomatic function ] → there exist

ϕ, ε = Null value

If specific something (meaningful symbol)




$C_2 g(n) \leq f(n) \leq C_1 g(n)$

[ $f(n)$ is the function that represents the time complexity of an algorithm ]

$f(n)$ → no of inputs, Initially, $f(n)$ is the function of the input.

$\Omega$ → smaller time complexity (at least) is Worst case

$\theta$ → average case

$O$ → Best case (at most)

[ $\Omega$ → Big Omega
$\omega$ → small omega ]

$A_1$ → $O(n)$ → At most time complexity

$\Omega(n)$ → At least time complexity

$A_1$ → $O(n^2)$ → max time req is $n^2$

$A_2$ → $O(n)$ → max time req is $n$

$A_1$ → $O(n)$ → $4n$

$A_2$ → $O(n)$ → $2n$

In term of Best case $A_2$ is better.

$A_1$ → $\Omega_1(n)$ → $n$ times req

$A_2$ → $\Omega_2(n)$ → $3n/2$ times req.

$\Omega_1(n) < \Omega_2(n)$

In terms of Worst case $A_1$ is better.

Threshold value of the function is a point from where the function is starting to stable.

[9/01/2023]

Fairt take a set of elements of 20 numbers then take the suggestive number from the given set. {99, 80, 13, 65, 11, 101, 25, 56, 78, 85
97, 30, 79, 86, 96, 47, 39, 110, 16, 19}

LAB

Suggestive → 65, 39, 57, 69

Step 1: take an array which contain 20 element we have to take a loop for 20 elements.

Step 2: Now we have to find the 1st number in the array so we have to compare the element with each element of the array. here also we have to take a for loop of 20 elements for the comparison. There the comparison will be continued until we find the element in the end of the array

---

10/Jan/2023

Pre-Requisites :-

1) Problem Solving → Creation and analysis of algorithm

Addition of 2 nos,

Step 1: take 2 nos and temp variable.

Step 2: Perform addition of 2 nos and store in temp variable.

Step 3: Output temp variable || Step 4: End

The logical step by step approach to solve a problem is called algorithm where the no. of steps must be finite.

Programs are set of instructions.

## Algorithm

① User friendly ie easy to understand.

② Machine independent.

③ Cannot be converted into machine code.

## Programming

① Not user friendly - not easy to understand.

② Machine dependent

③ Can be converted into machine code

TO DO

Q. What are the different complexities? What are the characteristic of algorithm?

Q. What are algorithm? Q. What are programs?

Q. What are the differences between program and algorithm.

- Asymptotic Notation :-

TO DO Q. What is asymptotic notation? Write the advantage of asymptotic notation. Write an example.

Q. An example of your own and find the space and time complexity of the care.

---

LAB

16/Jan/2023

## Insertion sorting

| 99 | 80 | 65 | 30 | 11 | 101 | 25 |
|---|---|---|---|---|---|---|

Algo: Comparison ① Compare first element with Second element. ie 90 with 80.

∴

| 80 | 90 | - - - |
|---|---|---|

Now 80 < 90 then a[i+1] = a[i]; and 80 will be stored in another element. and 80 will be stored in a[i];

Q.1. Find the least upperbound for $2n^2+n$. 06/Feb/2023

$\Rightarrow$ $c>0$ $|n>=$ threshold point $|$, Threshold $>=0$ $|$ $n>=1$

$f(n) = 2(n^2)+n = 2n^2+n <= c.g(n^2)$

$\Rightarrow 2n^2+n <= 2(n^2)$

$\Rightarrow 2n^2+n <= 3(n^2)$

$\Rightarrow n <= n^2$ divide both side by n

$\Rightarrow 1 <= n$

$\therefore$ $\boxed{n>=1}$

Q.2. Find the greatest lower bound for $2n^2+n$

Q.3. Find the average time complexity using above upper & lower bound example.

Ans② $f(n) = 2n^2+n = 2n^2+n >= c.g(n^2)$

$\Rightarrow 2n^2+n >= 3.n^2$

$\Rightarrow n >= n^2$

$\Rightarrow 1 >= n$ dividing both side with n.

$\therefore$ $\boxed{n<>1}$

- **Book Substitution :-**

$T(n)$ = Recurrence relation of binary search.

$T(n/2)+c$ if $n>1$

$1$ if $n=1$ // base prob. or sub prob.

$T(n) = T(n/2)+c$ ---- ①

$T(n/2) = T(n/4)+c$ ---- ⑪

$T(n/4) = T(n/8)+c$ ---- ⑪⑪

Now substitute eqn (ii) in (i) —

$T(n) = T(n/4) + c + c$

$= T(n/2^2) + 2c$

$= T(n/8) + c + 2c$

$= T(n/2^3) + 3c$

$= T(n/2^4) + 4c$

——- K times

$T(n/2^k) + kc$ we have to set $T(1) = 1$

Put $n/2k = 1$

$\Rightarrow n = 2^k$

$\Rightarrow \log n = \log 2^k$

$\Rightarrow k \log 2 = \log n$

$\Rightarrow k(1) = \log n$

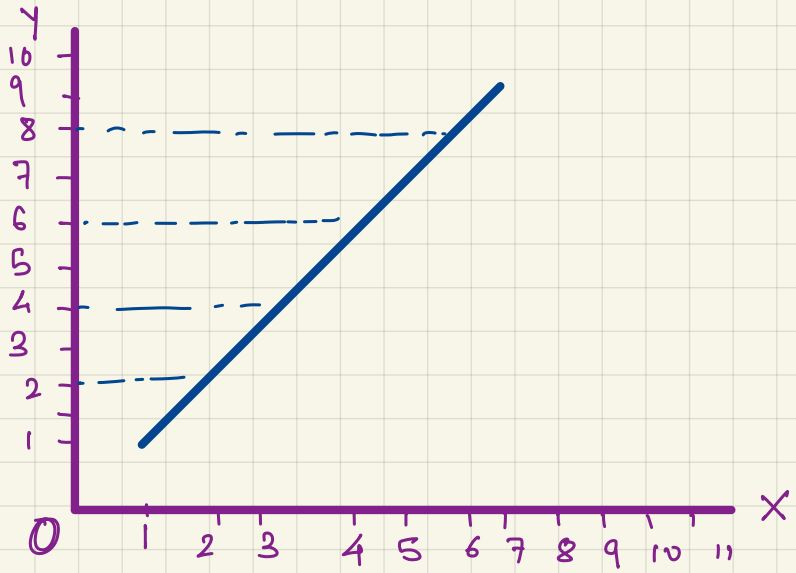
$\Rightarrow k = \log n$

$T(n/2^k) + k \cdot c$

$= T(1) + \log n \cdot c$

$= \log n \cdot c + 1$

$\therefore T(n) = O(\log n)$




order of growth

$y = 2x + 1$

$y = 3 \quad x = 1$

$y = 5 \quad x = 2$

$y = 7 \quad x = 3$

The time complexity doesn't depend on the input size n, irrespective of the input size the runtime will be always the same.

• General formula of Master theorem,

$\boxed{aT(n/b) + f(n)}$, where $a, b \geq 1$ and $f(n) = n^{\log_b a - \varepsilon}$ [take $\varepsilon = 1$]

① $T(n) = 16T(n/4) + n$

Soln: $\therefore a = 16$ and $b = 4$ and $f(n) = n^{\log_4 16 - \varepsilon}$

Now $\log_4 16 = \log_4 4^2$

$= 2\log_4 4 = 2 \cdot 1 = 2$

$\therefore f(n) = n^{2-1}$

$\Rightarrow f(n) = n$

$\Rightarrow n = O(n)$

Since the equation holds the 1st case of the master theorem applies to the given reccurence relation.

② $T(n) = 3T(n/2) + n$.

Soln: $a = 3$ and $b = 2$ and $n = f(n) = n^{\log_2 3 - \varepsilon}$

$\log_2 3 = \log_2 2 + 1$

$> \log_2 2 + \log_2 1$

$= 1 + 0 > 1$

$\therefore \Rightarrow f(n) = n^{1-1}$

$= n^0 = 1$

$n = 1 = O(1)$ → Constant time / Best case

③ $T(n) = 4T(n/2) + cn$

④ $T(n) = 4T(n/2) + n/\log n$

⑤ $T(n) = 4T(n/2) + \log n$

# Dynamic Programming

20/Feb/2021

## Matrix Chain :-

$A = \{4, 10, 3, 12, 20, 7\}$

**Step-1 :** Draw the matrix for i, j as

| J → | 1 | 2 | 3 | 4 | 5 | i ↓ |
|---|---|---|---|---|---|---|
| | 0 | 120 | 264 | 1080 | | 1 |
| | | 0 | | | | 2 |
| | | | 0 | | | 3 |
| | | | | 0 | | 4 |
| | | | | | 0 | 5 |

**Step-2 :** Assign product numbering.

| 4 | 10 | 3 | 12 | 20 | 7 |
|---|---|---|---|---|---|
| $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ |

**Step-3 :** Check for i, j

$i \leq k < j$

i : 1. 3

2.

if $i == j$, then result value = 0

if $i < j$

$do\{ m[i,j] = \min [i,k] + n[k+1, j] + P_{i-1} \cdot P_j \cdot P_k$

$i = 1, j = 2, k = 1$

$m[1,2] = \min[1,1] + n[2,2] + P_0 P_2 P_1$

$= 4 \times 3 \times 10 = 120$

$i = 1, j = 3, k = 2$

$m[1,3] = \min [1,2] + n[3,3] + P_0 P_3 P_2$

$= 120 + 4 \times 12 \times 3 = 264$

$$\min[1,4] = \min \begin{cases} m[1,3] + n[4,4] + P_0 P_3 P_4 = 264 + 0 + 4 \times 10 \times 20 = 1224 \\ m[1,2] + n[3,4] + P_0 P_2 P_4 = 120 + 120 + 4 \times 3 \times 20 = 1080 \\ m[1,1] + n[2,4] + P_0 P_1 P_4 = 2120 \end{cases}$$

$\therefore \min[1,4] = 1080$

# Strassen's Algorithm :-

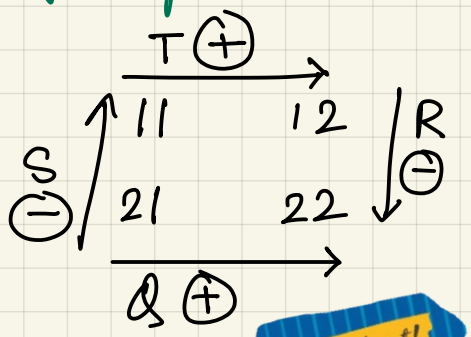
$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \quad C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

## Formula :-

- $P = (A_{11} + A_{22})(B_{11} + B_{22})$ → product form of summation of diagonal element.
- $Q = B_{11}(A_{21} + A_{22})$
- $R = A_{11}(B_{12} - B_{22})$
- $S = A_{22}(B_{21} - B_{11})$
- $T = B_{22}(A_{11} + A_{12})$

REMEMBER

| B | A | A | B |
|---|---|---|---|
| 11 | 11 | 22 | 22 |




Important!

- $U = (A_{21} - A_{11})(B_{11} + B_{12})$ → [written in form of ST, but As are converted to Bs and Bs to As]
- $V = (B_{21} + B_{22})(A_{12} - A_{22})$ → [written in QR form but As are converted to Bs and Bs are converted to As]

$C_{11} = P + S - T + V$

$C_{12} = R + T$

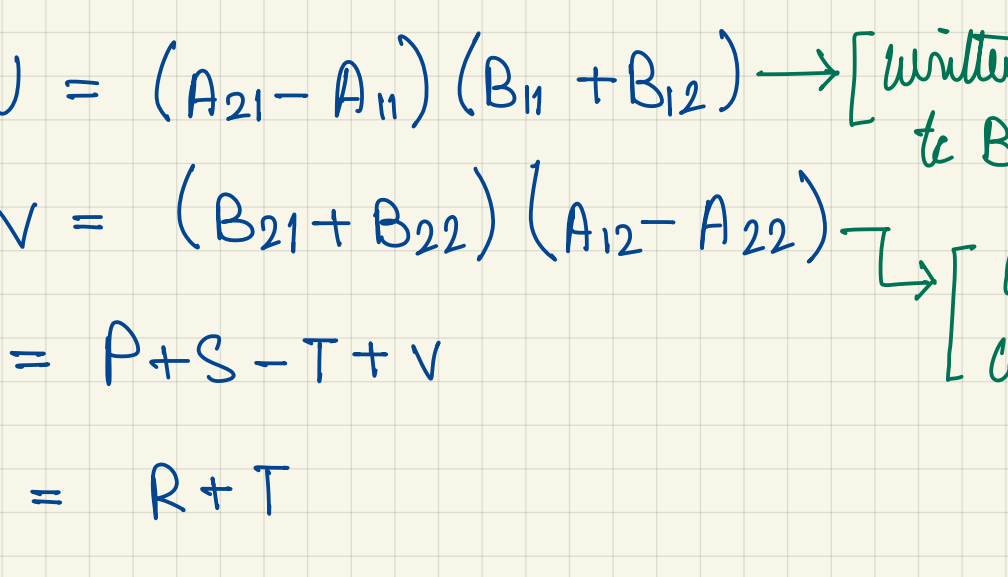
$C_{21} = Q + S$

$C_{22} = P + R - Q + U$

---

$X = (B, A, C, D, B)$ $Y = (B, D, C, B)$ Ans: BCB

22/Feb/2023




Rules :-

① When X = Y, put diagonal arrows ↖

② Wherever there is diagonal arrow, add 1 to the diagonal element and put in present cell.

③ When there is no diagonal arrow, compare with the top right diagonal element and put the greater element and put arrow pointing the greater element. and when the elements are equal put upward arrow. ↑

* is the answer. ↖

④ Start traversing from bottom right element, follow the arrows, the index of diagonal arrows *

Q.1. What is Dijsktra's algorithm? Real life example.

Q.2. Implementation of Dijkstra algorithm.

Dijkstra's algorithm is a graph traversing algorithm used to find the shortest path between two nodes in weighed graph. Dijkstra's algorithm is a simple modification of breadth first search. The algo maintain a set of visited nodes and set of unvisited nodes and compute the shortest path to every other node in the graph. At each iteration the algo choose the universal node with the shortest distance from the starting node and visit all its neighbour. Dijkstra algorithm works only for graph, with the non negative edge weights.