# DAA ASSIGNMENT#01

Saturday, 6 September 2025 5:55 PM

NABIRA KHAN 23K-0914

## QUESTION #01

ALGORITHM → for i=0 to m-1 outer loop
for j=0 to n-1 inner loop
c[i][j] = A[i][j] + B[i][j] addition + assignment
return C

| STATEMENT | OP TYPE | TIMES | COST SYM |
|---|---|---|---|
| i=0 | assign | 1 | a |
| i<m-1 | compare | m | b |
| i++ | increment | m | c |
| j=0 | assign | 1 | d |
| j<n-1 | compare | n | e |
| j++ | increment | n | f |
| C[i][j], A, B | offset calc | mn | g |
| addition | add + assign | mn | h |

$T(n) = a + bm + cm + d + en + fn + gmn + hmn$
$= (a+d) + m(b+c) + n(e+f) + mn(g+h)$
$= c_1 + mc_2 + nc_3 + mnc_4$
$= \boxed{O(mn)}$

$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$ $B = \begin{bmatrix} 2 & 1 & 4 & 3 \\ 8 & 7 & 6 & 5 \\ 12 & 11 & 10 & 9 \end{bmatrix} \longrightarrow C = \begin{bmatrix} 3 & 3 & 7 & 7 \\ 13 & 13 & 13 & 13 \\ 21 & 21 & 21 & 21 \end{bmatrix}$

TRACE TABLE

| i | j | A[i][j] | B[i][j] | C[i][j] = A[i][j]+B[i][j] |
|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 |
| 0 | 1 | 2 | 1 | 3 |
| 0 | 2 | 3 | 4 | 7 |
| 0 | 3 | 4 | 3 | 7 |
| 1 | 0 | 5 | 8 | 13 |
| 1 | 1 | 6 | 7 | 13 |
| 1 | 2 | 7 | 6 | 13 |
| 1 | 3 | 8 | 5 | 13 |
| 2 | 0 | 9 | 12 | 21 |
| 2 | 1 | 10 | 11 | 21 |
| 2 | 2 | 11 | 10 | 21 |
| 2 | 3 | 12 | 9 | 21 |

## QUESTION #02

ALGORITHM → for i=0 to n-1
if A[i] == key then
return i
return -1

traverses through entire array till key found via comparison or till array ends.

| i | A[i] | Compare A[i] == 32 | Found? |
|---|---|---|---|
| 0 | 12 | 12==32 → No | No |
| 1 | 27 | 27==32 → No | No |
| 2 | 19 | 19==32 → No | No |
| 3 | 32 | 32==32 → Yes | Yes |

OUTPUT → index = 3
Best case → O(1)
Worst case → O(n)
Avg case → O(n)

## QUESTION #03

$100n^2 < 2^n$

$n = 10 \rightarrow 10000 < 1024$ ✗
$n = 14 \rightarrow 19600 < 16384$ ✗
$n = 15 \rightarrow 22500 < 32768$ ✓

Smallest $n = 15$ for which $100n^2$ works faster than $2^n$

## QUESTION # 04

a)

```
Algorithm Fun(n)
        Sum=0;
        For(i=n^2; i>=1 ; i/2)
                Sum=sum+I
                Printf("The Value of Sum is %d", sum)
```

i decreases exponentially (divide by 2 each iteration) can be shown directly as O(log n)
let T = number of times loop body execute

start → $i_0 = n^2$ while $i \geq 1$, halving each iteration
pattern of i → $n^2, \frac{n^2}{2}, \frac{n^2}{4} \ldots$

$\therefore \frac{n^2}{2^k} = 1$

$n^2 = 2^k$ hence → $\boxed{T(n) = O(\log n)}$

$\log_2 n^2 = \log_2 2^k$

$2\log_2 n = k$

b)

```
Algo fun(n)
        int i, j, k, p, q = 0
        for(i=1; i<n ; i++)
                P=0;
                For(j=n;j>1;j=j/2)
                        ++p;
                For(k=1;k<p;k=k*2)
                        ++q
        return q;
```

loop patterns

→ i increases linearly by 1 (outer) → $O(n)$

→ j decreases exponentially (divide by 2) (inner)

→ k increases exponentially (doubling) (inner)

FOR EACH OUTER ITERATION

$j \rightarrow \log_2 n = O(\log n)$ ∴ sets $p = O(\log n)$ pattern → $n, \frac{n}{2}, \frac{n}{4} \cdots \frac{n}{2^k} \rightarrow \log_2 n = k$

$k \rightarrow \log_2 p = O(\log p)$, since $p = O(\log n)$ pattern → $2^k = p \rightarrow k = \log_2 p$

$\log p = O(\log \log n)$

∴ per outer iteration, cost → $O(\log n + \log\log n) = O(\log n)$

outer loop (i) runs n-1 times → $O(n)$ times

$\boxed{T(n) = O(n \log n)}$

c)

```
while(m!=n)
        if(m>n)
                m=m-n
        else
                n=n-m
```

→ both m and n decrease overtime (smaller subtracted from the larger at each iteration till m=n)

→ worst case: one of them is 1 → if $n=1$, $m=M$ → $M-1$ iterations → $O(M)$

→ if $m=1$, $n=N$ → $N-1$ iterations → $O(N)$

→ best case: m = n initially, 0 iterations

∴ $\boxed{T(m,n) = O(\max(m,n))}$

d)

```
algo fun(n)
        int i, j, k=0;
        for(i=n/2;i<=n;i++)
                for(j=2;j<=n; j=j*2)
                        k=k+n/2
        return k:
```

→ i increases linearly by 1 while $i \le n$ → $O(n)$ (outer) pattern → $\frac{n}{2}, \frac{n}{2}+1, \frac{n}{2}+2 \cdots$

→ j increases exponentially (doubling) while $j \le n$ → $O(\log n)$ (inner) pattern → $2, 4, 8 \cdots \rightarrow 2^k = n$

$k = \log_2 n$

work per inner iteration constant → $O(1)$

∴ $\boxed{T(n) = O(n \log n)}$

e)

```
k=1;
for(i=0; i<n; i++)
        for(j=0; j<n; j=j+k)
                printf("%d \t", j);
        k=k*2;
```

$k = k*2 \rightarrow O(1)$ constant

→ i increases linearly by 1 while $i<n$ → $O(n)$ (outer) pattern → $0, 1, 2 \cdots n$

→ j increases by k which increases exponentially (doubling each outer iteration) (inner) pattern →

$k=1 \rightarrow j = 0, 1, 2 \cdots n-1 \rightarrow O(n)$

$k=2 \rightarrow j = 0, 2, 4 \cdots n-1 \rightarrow O(\frac{n}{2})$

$k=4 \rightarrow j = 0, 4, 8 \cdots n-1 \rightarrow O(\frac{n}{4})$

→ this means since $j=0, j=k, j=2k$ until $<n$

∴ this is a geometric progression → $n + \frac{n}{2} + \frac{n}{4} \cdots$

$\boxed{T(n) = O(n)}$

**QUESTION #05**

**PART A**

① $5n^2 - 100n + 50 \in O(n^2)$

$5n^2 - 100n + 50 \le cn^2$ for $n \ge n_0$

taking absolute values + dominant term bounding approach

$100n \le 100n^2$, $50 \le 50n^2$ for $n \ge 1$

$5n^2 - 100n + 50 \le 5n^2 + 100n^2 + 50n^2$

$5n^2 - 100n + 50 \le 155n^2$

$5n^2 - 100n + 50 - 155n^2 \le 0$

$-150n^2 - 100n + 50 \le 0$

$150n^2 + 100 - 50 \ge 0$ → solve quadratic

$\boxed{c = 155, \quad n_0 = 1}$ proved!

② $n^2 + n\log n \in O(n^2)$

$n^2 + n\log n \le cn^2$ for $n \ge n_0$

$n\log n \le n^2$

② $n^2 + n\log n \in O(n^2)$

$n^2 + n\log n \le cn^2$ for $n \ge n_0$

$n\log n \le n^2$

$n^2 + n\log n \le n^2 + n^2$

$n^2 + n\log n \le 2n^2$

$n\log n \le n^2$

$\log n \le n$ for all $n \ge 1$

$\boxed{c = 2, \ n_0 = 1}$ proven!

③ $n(\log n)^2 + n\log n \in O(n(\log n)^2)$

$n(\log n)^2 + n\log n \le cn(\log n)^2$ for $n \ge n_0$

$n\log n \le n(\log n)^2$

$n(\log n)^2 + n\log n \le n(\log n)^2 + n(\log n)^2$

$n(\log n)^2 + n\log n \le 2n(\log n)^2$

$\cancel{n}\log n \le \cancel{n}(\log n)^2$

$\log n \le (\log n)^2$

$0 \le (\log n)^2 - \log n$

$0 \le \log n(\log n - 1)$

$\log n \le 0$ $\quad \log n - 1 \ge 0$

$\hookrightarrow n \le 1$ $\quad \log n \ge 1$

$\hookrightarrow n \ge b^1$ $\quad \log_2 2 \ge 1$

base $= 2$

$\boxed{c = 2, \ n_0 = 2}$ proven!

④ $n^4 + 50n^3 \notin O(n^3)$

$n^4 + 50n^3 \le cn^3$ for $n \ge n_0$

$n^4 > n^3$ for all $n$

$f(n) >$ degree of $g(n)$, $f(n) \notin O(g(n))$

**PART B**

⑤ $4n^2 - 1000n + 25 \in \Omega(n^2)$

$4n^2 - 1000n + 25 \ge cn^2$ for $n \ge n_0$

using division by dominant term approach

$\lim_{n\to\infty}\left(\frac{4n^2 - 1000n + 25}{n^2}\right) = 4 - \frac{1000}{n} + \frac{25}{n^2}$

as $n \to \infty$ result $= 4$

$L = 4$ is $> 0$ hence $4n^2 - 1000n + 25 \in \Omega(n^2)$ proven!

$0 < c \le 4 \longrightarrow c = 2$

$4n^2 - 1000n + 25 \ge 2n^2$

$2n^2 - 1000n + 25 \ge 0 \longrightarrow$ solve quadratic

$\boxed{c = 2, \ n_0 = 500}$

⑥ $n^2 + n\log n \in \Omega(n^2)$

$n^2 + n\log n \ge cn^2$ for $n \ge n_0$

$\lim_{n\to\infty}\left(\frac{n^2 + n\log n}{n^2}\right) = 1 + \frac{\log n}{n}$

as $n \to \infty$ result $= 1$

$L = 1$ is $> 0$ hence $n^2 + n\log n \in \Omega(n^2)$ proven!

$0 < c \le 1 \longrightarrow c = 1$

$n^2 + n\log n \ge n^2$

$n\log n \ge 0$

$\boxed{c = 1, \ n_0 = 1}$

⑦ $\log n \notin \Omega(n)$

$\log n \ge cn$ for $n \ge n_0$

divide both sides by $n$

$\frac{\log n}{n} \ge c$

as $n \to \infty$, result $= 0$

hence $\log n \notin \Omega(n)$ + $\log n$ alway $< n$

**PART C**

⑧ $10n^2 - 200n + 500 \in \Theta(n^2)$

$c_1 n^2 \le 10n^2 - 200n + 500 \le c_2 n^2$ for $n \ge n_0$

using divide by dominant term approach

$\lim_{n\to\infty}\left(\frac{10n^2 - 200n + 500}{n^2}\right) = 10 - \frac{200}{n} + \frac{500}{n^2} = 10$

$0 < L < \infty$ since $L = 10$ hence $10n^2 - 200n + 500 \in \Theta\, n^2$

proven!

taking $c_1 = 9$, $c_2 = 11$

$10n^2 - 200n + 500 \ge 9n^2$

$n^2 - 200n + 500 \ge 0 \longrightarrow$ solve quadratic

$n_0 = 201$

$10n^2 - 200n + 500 \leq 11n^2$

$n^2 + 200n - 500 \geq 0 \longrightarrow$ solve quadratic

$n_0 = 500$

$\therefore \boxed{c_1 = 9,\ c_2 = 11,\ n_0 = 200}$

(9) $n^2 + n\log n \in \Theta(n^2)$

$c_1 n^2 \leq n^2 + n\log n \leq c_2 n^2$ for $n \geq n_0$

$\lim_{n\to\infty}\left(\frac{n^2 + n\log n}{n^2}\right) = 1 + \frac{\log n}{n} = 1$

$0 < L < \infty$ since $L = 1$ hence $n^2 + n\log n \in \Theta(n^2)$

proven!

taking $c_1 = 1$ and $c_2 = 2$

$n^2 + n\log n \geq n^2$

$n\log n \geq 0 \longrightarrow n \geq 1$

$n^2 + n\log n \leq 2n^2$

$n\log n \leq n^2$

$0 \leq n(n - \log n)$

$n \geq 0 \qquad n - \log n \geq 0$

$n \geq \log n \qquad n \geq 1$

$\boxed{c_1 = 1,\ c_2 = 2,\ n_0 = 1}$

(10) $n\log n + 50 \in \Theta(n\log n)$

$c_1 n\log n \leq n\log n + 50 \leq c_2 n\log n$ for $n \geq n_0$

$\lim_{n\to\infty}\left(\frac{n\log n + 50}{n\log n}\right) = 1 + \frac{50}{n\log n} = 1$

$0 < L < \infty$ since $L = 1$ hence $n\log n + 50 \in \Theta\ n\log n$

proven!

taking $c_1 = 1$ and $c_2 = 2$

$n\log n + 50 \geq n\log n$

$50 \geq 0$ holds for all $n \longrightarrow$ taking $n_0 = 1$

$n\log n + 50 \leq 2n\log n$

$50 \leq n\log n \longrightarrow n_0 = 14$

$\boxed{c_1 = 1,\ c_2 = 2,\ n_0 = 14}$