Date.

Design and Analysis of Algorithm
Assignment 1

ARSH - BAI-5A
23K-0078

Q1: MatrixAddition (A, B, m, n)

```
for i = 0 to m-1                      ->  m
    for j = 0 to n-1                  ->  m x n
        C[i][j] = A[i][j] + B[i][j]       m(n-1)

return C;
```

Time function = $(m) + (m \times n) + (m \times (n-1))$

$= m + mn + mn - m$

$= 2(mn)$

Time Complexity = $O(mn)$ or $O(n^2)$

Trace for 3x4 matrix.

let,

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

Next Page →

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	3	5
0	2	3	4	7
0	3	4	5	9
1	0	5	6	11
1	1	6	7	13
1	2	7	8	15
1	3	8	9	17
2	0	9	10	19
2	1	10	11	21
2	2	11	12	23
2	3	12	13	25

$$C = \begin{bmatrix} 3 & 5 & 7 & 9 \\ 11 & 13 & 15 & 17 \\ 19 & 21 & 23 & 25 \end{bmatrix}$$

Q2: LinearSearch(A, n, key)

Array of size n and key/value to search

```
For i = 0 to n                    → Start a loop from 0 index.
    if A[i] = key.                → Compare current with target.
        return i  // found.       → if match return index
return -1  // not found.          → return -1 if not found.
```

Trace search for 8

A = [2, 4, 6, 8, 10]

Date.

i	A[i]	Comparision	result.
0	2	$2 \neq 8$	Continue
1	4	$4 \neq 8$	Continue.
2	6	$6 \neq 8$	Continue
3	8	$8 = 8$	found → return 3

Q3: $100n^2 < 2^n$.

Log on b.s.

$$\log(100n^2) < \log 2^n$$

$$\log_2 100 + 2\log_2 n < n \log_2 2$$

$$n > \log_2 100 + 2\log_2 n$$

for $n=10$ $10 > 6.64 + 2\log_2 10 = 10 > 13.28 \rightarrow$ ✗

for $n=15$ $15 > 6.64 + 2\log_2 15 = 15 > 14.45 \rightarrow$ ✓

for $n=14$ $14 > 6.64 + 2\log_2 14 = 14 > 14.1 \longrightarrow$ ✗

Hence $n = 15$

Q4:

Algorithm fun(n)

Sum = 0;

for (i = n^2 ; i >= 1 ; i/2)

sum = sum + i

Printf ("The Value of Sum is %d", sum)

$n = 8$

i	Sum	Pattern	Assume $i < 1$
64	1	n^2	$i = n^2/2^k$
32		$n^2/2$	$n^2/2^k < 1$
16		$n^2/2^2$	$n^2 < 2^k$
⋮			
1		$n^2/2^k$	$\log n^2 < \log 2^k$

$2 \log_2 n < k \log_2 2$

$k = 2 \log n \rightarrow$ Time Complexity.
$O(\log n)$

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;
```

inner $= O(\log n + \log(\log n))$
$= O(\log n)$

Outer $= O(n)$ ∴ linear.

∴ Time Complexity $= O(n \log n)$

Pattern is increasing since linear outer loop and logarithmic inner loop.

Date.

loop i : runs for n times.
loop j : $j < 1$

$\frac{n}{2^k} < 1$

$\log n < \log 2^k$

$\boxed{k = \log n}$ ⟶ hence $p = \log n$

loop k (inner) $p = \log n$.

$k > p$

$2^k > \log n$

$k \log_2 2 > \log(\log n)$

$\therefore k = \log(\log n)$

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

assume $n \geq m$

n	m
n = 10	m = 2
10	2
8	2
6	2
4	2
2	2

$O(\frac{n}{m})$ time ⟶ Larger value of n and m will come in numerator and other in denominator

$\therefore$ Time Complexity $= O(n)$
or $O(\max m, n)$
Best case : $O(1)$

$\therefore$ Pattern is decreasing values of m or n depending on which one is Larger.

Date.

d) algo fun(n):

```
int i, j, k=0
for (i=n/2 ; i<n; i++) -> n+1
    for (j=2 ; j<=n ; j=j×2) -> n logn
        K=(K+n)/2 -> n(log n)
return k; -1
```

Outer loop = $n/2 \rightarrow \therefore$ n times : $O(n)$

inner loop = $j > n$

$2^k > n$

$k \log_2 2 > \log n$

$k = \log n$

$= O(\log n)$

Timefunction $= (n+1) + n\log n + n \log n$

$TC = O(n \log n)$

$\therefore$ Time complexity $= O(n \log n)$

pattern increasing as outer loop $\frac{n}{2} \rightarrow n$ and inner loop exponentially by 2^k

(e) K=1

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

n=10

i	j	k
0	0	1
1	⋮	1
0	10	1
1	0	2
1	2	2
1	10	2
2	0	4

Date.

K is increasing exponentially : 1, 2, 4, ---- 2^n

$\frac{n}{K}$ ← J is decreasing : $n + \frac{n}{2} + \frac{n}{4}$, $\frac{n}{2^m}$ / $\frac{n}{2^K}$

$T(n) = n + \frac{n}{2} + \frac{n}{4} + \frac{n}{8} \cdots \frac{n}{2^m}$ → Total time inner + outer

$T(n) \leq 2n$

$\therefore O(n)$

→ Pattern is decreasing since K is increasing exponentially by 2^K

Q5 (a) Proofs of Big O Upper Bound.

1. $\underbrace{5n^2 - 100n + 50}_{f(n)} \in \underbrace{O(n^2)}_{cg(n)}$

$f(n) \leq c \cdot g(n)$

$\frac{f(n)}{gn} \leq c \rightarrow \frac{5n^2 - 100n + 50}{n^2} \leq c$

$5 - \frac{100}{n} + \frac{50}{n^2} \leq c$

$\lim_{n \to \infty} \frac{f(n)}{g(n)} \rightarrow \therefore 5 \leq c$

$c \geq 5$

$c = 6$

$5 - \frac{100}{n} + \frac{50}{n^2} \leq 6$

$n^2 \times \left(-\frac{100}{n} + \frac{50}{n^2} \right) \leq 1 n^2$

$-100n + 50 \leq n^2$

$n^2 + 100n - 50 \geq 0$

$n = 1$ & $c = 6$

$5n^2 - 100n + 50 \leq 6n^2$ $\therefore O(n^2)$ proved

Date.

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

$$\frac{f(n)}{g(n)} \leq c$$

$$\frac{n^2 + n\log n}{n^2} \leq c$$

$$1 + \frac{\log n}{n} \leq c$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} \longrightarrow 1 \leq c$$

$$c \geq 1$$

$c = 2$ $\quad 1 + \frac{\log n}{n} \leq 2$

$$\frac{\log n}{n} \leq 1$$

$\log n \leq n$ for all $n \geq 1$

for $n = 1$: $\log(1) = 0 \leq 1$
for $n = 2$: $\log(2) = 0.69 \leq 2$ $\quad \in$ inequality hold.
hence proved $c \geq 2$ & $n \geq 1$

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

$$n(\log n)^2 + n\log n \leq c \cdot n(\log n)^2, \ \forall n > n_0$$

$$(\log n)^2 + \log n \leq c(\log n)^2$$

Divide b.s by $\log n$: $\log n + 1 \leq c \cdot \log n$.

$$1 + \frac{1}{\log n} \leq c$$

Hence $n_0 = 2$, $c = 2$.

Date.

$$1 + \frac{1}{\log n} \leq 2$$

$$1 \geq \frac{1}{\log n}$$ ∴ This inequality holds for $n \geq 2$.

Therefore, $n(\log n)^2 + n \log n \in O(n(\log n)^2)$

(a) $n^4 + 50n^3 \in O(n^3)$

$$\frac{n^4 + 50n^3}{n^3} \leq c$$

$$n + 50 \leq c$$

as $n \to \infty$, no matter how large the constant c we pick it would be insufficient.

∴ no value of c can be larger than $n+50$ for all $n \geq n_0$.

Therefore $n^4 + 50n^3 \notin O(n^3)$

(b) I. $4n^2 - 1000n + 25 \in \Omega(n^2)$

$$4n^2 - 1000n + 25 \geq c \cdot n^2$$

$$\frac{f(n)}{g(n)} \leq c$$

$$\frac{4n^2 - 1000n + 25}{n^2} \geq c$$

$$4 - \frac{1000}{n} + \frac{25}{n^2} \geq c$$

$$\lim_{n \to 100} \frac{f(n)}{g(n)} \Rightarrow 4 \geq c$$

$$c \leq 4$$

Date

$c = 3$

$$4 - \frac{1000}{n} + \frac{25}{n^2} \geq 3$$

$$n^2\left(-\frac{1000}{n} + \frac{25}{n^2}\right) \geq (-1) n^2$$

$$-1000n + 25 \geq -n^2$$

$$n^2 + 25 - 1000n \leq 0$$

as n becomes greater (n^2) will dominate $-1000n$

$n \geq 21$

$c \leq 4$ Therefore: $4n^2 - 1000n + 25 \in \Omega(n^2)$

6. $n^2 + n\log n \in \Omega(n^2)$

$n^2 + n\log n \geq c \cdot n^2$ $n > n_0$.

$$\frac{f(n)}{g(n)} \geq c$$

$$\frac{n^2 + n\log n}{n^2} \geq c$$

$$1 + \frac{\log n}{n} \geq c$$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} \Rightarrow 1 \geq c$$

$c \leq 1$

$c = 1$

$$1 + \frac{\log n}{n} \geq 1$$

for $n = 1$: $\log(1) \Rightarrow 0 \geq 0$

for $n = 2$: $\log(2) = 1 \geq 0$

$$\frac{\log n}{n} \geq 0$$

$$\log n \geq 0$$

proved for $n \geq 1$ and $c = 1$

Date.

7. $\log n \in \Omega(n)$

$\log n \geq c \cdot n \quad \forall\, n \geq n_0$

divide by n b.s.

$\frac{\log n}{n} \geq c \rightarrow$ as $n \rightarrow \infty$: $\frac{\log n}{n} = 0$

$\therefore \quad \log n \notin \Omega(n)$

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

$c_1 n^2 \leq 10n^2 - 200n + 500 \leq c_2 n^2$

for upper bound,

$10n^2 - 200n + 500 \leq c_2 (n^2)$

$\frac{f(n)}{g(n)} \leq c_2 \quad \rightarrow \quad \frac{10n^2 - 200n + 500}{n^2} \leq c_2$

$10 - \frac{200}{n} + \frac{500}{n^2} \leq c_2$

$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} \Rightarrow 10 \leq c_2$

$c_2 > 10.$

$c_2 = 11$

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

$n\left(-\frac{200}{n} + \frac{500}{n^2}\right) \leq (1)\, n^2$

$-200n + 500 \leq n^2 \rightarrow n^2 + 200n - 500 \leq 0$

$n = 2.67$

$\boxed{n = 3 \;,\; c_2 \geq 10}$ upper bound

Date.

Lower bound: $10n^2 - 200n + 500 \geq c_2(n^2)$

$$\frac{f(n)}{g(n)} \geq c_1 \rightarrow \frac{10n^2 - 200n + 500}{n^2} \geq c_1$$

$$10 - \frac{200}{n} + \frac{500}{n^2} \geq c_1$$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} \rightarrow 10 \geq c_2 \longrightarrow c_2 \leq 10$$

taking $c_2 = 9$.

$$10 - \frac{200}{n} + \frac{500}{n^2} \geq 9.$$

$$\frac{-200n + 500}{n^2} \geq -1$$

$$-200n + 500 \geq -n^2$$

$$n^2 - 200n + 500 \geq 0$$

greater $\Leftarrow$ $n = 200$, $c \leq 10$

$c_1 = 9$, $c_2 = 10$, $n = 200$.

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

$c_1 n^2 \leq n^2 + n\log n \leq c_2 n^2$

Lower bound: $n^2 + n\log n \geq n^2$

So $c_1 = 1$, $n_0 = 1$

Upper bound: $n^2 + n\log n \leq c_2 n^2$

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

$c = 2$, $n_0 = 2$

$1.n \leq n^2 + n\log n \leq 2n^2$

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

Date

10. $n \log n + 50 \in n \log n$

$C_1 n \log n \le n \log n + 50 \le C_2 n \log n$

Upper bound:

$$n \log n + 50 \le C_2 n \log n$$

$$1 + \frac{50}{n \log n} \le C_2$$

As $n \to \infty$: $\frac{50}{n \log} \to 0$ so LHS. Approches 1

Thus $C_2 = 2$ & $n_0 = 10$

Lower bound: $n \log n + 50 \ge c_1 n \log n$

$$1 + \frac{50}{n \log n} \ge c_1$$

so $c = 1$, $n_0 = 2$

$n \log n + 50 \in \Theta(n \log n)$, max $(2, 10)$ -