Ibrahim Johar Farooqi
BAI-5A
23K-0074

Date ____ 20__

DAA – Assignment 01

Q.1) design an algo for 2D matrix addition. compute its time complexity using frequency count method. also trace algorithm, for the array of size 3x4

↳
```
for(i=0; i<n; i++)          ⟶ n+1
{                                          ← loop
    for(j=0; j<n; j++)      ⟶ n x (n+1)
    {
        Arr[i,j] = A[i,j] + B[i,j];   ⟶ n x n
    }
}
return Arr;                 ⟶ 1
```
(i=0: 1 ; i<n: n+1 ; i++: n)

↳ $f(n) = (n+1) + (n(n+1)) + (n \times n) + 1$
$= n+1+n^2+n+n^2+1$
$= 2n^2+2n+1$

↳ time complexity: $O(n^2)$

↳ trace – 3x4 matrix addition

↳ matrix A:
$$= \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

↳ matrix B:
$$= \begin{bmatrix} 2 & 0 & 1 & 3 \\ 4 & 1 & 0 & 5 \\ 6 & 2 & 3 & 1 \end{bmatrix}$$

↳ i=0

A B
① ↳ j=0 → 1+2 = 3 → Arr[0][0] = 3
② ↳ j=1 → 2+0 = 2 → Arr[0][1] = 2
③ ↳ j=2 → 3+1 = 4 → Arr[0][2] = 4
④ ↳ j=3 → 4+3 = 7 → Arr[0][3] = 7

Date _____ 20___

↳ i=1

		A	B	
(5)	↳ j=0	5	4	5+4=9 → Arr[1][0]=9
(6)	↳ j=1	6	1	6+1=7 → Arr[1][1]=7
(7)	↳ j=2	7	0	7+0=7 → Arr[1][2]=7
(8)	↳ j=3	8	5	8+5=13 → Arr[1][3]=13

↳ i=2

		A	B	
(9)	↳ j=0	9	6	9+6=15 → Arr[2][0]=15
(10)	↳ j=1	10	2	10+2=12 → Arr[2][1]=12
(11)	↳ j=2	11	3	11+3=14 → Arr[2][2]=14
(12)	↳ j=3	12	1	12+1=13 → Arr[2][3]=13

↳ final matrix:

$$Arr = \begin{bmatrix} 3 & 2 & 4 & 7 \\ 9 & 7 & 7 & 13 \\ 15 & 12 & 14 & 13 \end{bmatrix}$$

(Q.2) Linear Search:

↳ Arr: 12, 27, 19, 32, 45

↳ Key: 32

↳ starting from the first element (i=0)

↳ compare the curr element w/ key (32)

↳ if element is equal to target key, return index i

↳ else, go to next element (i=i+1)

↳ repeat above steps until key is found

↳ if the end of array is reached w/o finding the key then return -1.

Date _____ 20___

↳ LinearSearch (Arr, n, key)

```
{
    for (i=0; i<n; i++)
    {
        if (Arr[i] == key)
            return i;
        return -1;
    }
}
```

↳ Time Complexity : O(n)

↳ Trace ; Arr: 12, 27, 19, 32, 45 , n=5 , key=32

↳ i=0

Arr[0] = 12 ; 12 ≠ 32 ; i = i+1;

↳ i=1

Arr[1] = 27 ; 27 ≠ 32 ; i = i+1;

↳ i=2

Arr[2] = 19 ; 19 ≠ 32 ; i = i+1;

↳ i=3

Arr[3] = 32 ; 32 == 32 ; return i=3 , as key is found.

Q.3) the smallest value of n, such that an algo whose running time is $100n^2$, runs faster than an algo whose running time is 2^n on same machine

↳ $100n^2 < 2^n$

↳ n = 5

↳ $100(5)^2 = 2500$ ↳ $2^5 = 32$

↳ $100n^2 > 2^n$

↳ n = 10

↳ $100(10)^2 = 10000$ ↳ $2^{10} = 1024$

↳ $100n^2 > 2^n$

Date _____ 20___

↳ n=12
↳ $100(12)^2 = 14400$ ↳ $2^{12} = 4096$
↳ $100n^2 > 2^n$

↳ n=14
↳ $100(14)^2 = 19600$ ↳ $2^{14} = 16384$
↳ $100n^2 > 2^n$

↳ n=15
↳ $100(15)^2 = 22500$ ↳ $2^{15} = 32768$
↳ $100n^2 < 2^n$

↳ **n=15**, smallest val for which the $100n^2$ algo is faster

Q.4) a) Algorithm Fun(n)

```
sum = 0;
for(i = n^2; i >= 1; i/2)
{
    sum = sum + i;
    print("value of sum is %d", sum);
}
```

↳ using n=8

	i	sum	step
↳	64	0+64=64	$n^2/2$
↳	32	64+32=96	$n^2/2^2$
↳	16	96+16=112	$n^2/2^3$
↳	8	112+8=120	$n^2/2^4$
↳	4	120+4=124	$n^2/2^5$
↳	2	124+2=126	$n^2/2^6$
↳	1	126+1=127	$n^2/2^7$

$\downarrow$

$n^2/2^k$

Date ____ 20 ____

↳ assume $i < 1$

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

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

$$n^2 = 2^k$$

$$2 \log n = \log_2 (2^k)$$

$$2 \log n = k$$

$$\boxed{k = 2 \log n}$$

↳ time complexity : $\boxed{O(\log n)}$

↳ pattern is Decreasing, halved after every iteration

b) Algo fun(n)

```
int i, j, k, p, q = 0;
for(i=1; i<n; i++){          ⟶ loop runs (n) times
  p=0;
  for(j=n; j>1; j=j/2){      ⟶ inner loop1 runs (log n) times
    p++;
  }
  for(k=1; k<p; k=k*2){      ⟶ inner loop2 (doubles each time)
    q++;                          log p
  }                  (k<p: log n)
}
return q;
```

↳ inner loop 1 runs log n times

$$n/2^k \leq 1$$

$$n = 2^k$$

$$\log n = \log_2 (2^k) \rightarrow k = \log_2 n$$

↳ hence $\boxed{p = \log n}$

↳ inner loop 2

↳ $p = \log n \rightarrow$ num of iterations $= \log p$

↳ $k \geq p$

$2^k \geq \log p$

$k \log_2 2 = \log p$

$\boxed{k = \log (\log n)}$

QUALITY PAPER PRODUCTS

Date _____ 20__

↳ inner loop: $O(\log n) + O(\log(\log n))$
$= O(\log n + \log(\log n))$
$= O(\log n)$

↳ outer loop = n

↳ **time complexity = $O(n \log n)$**

↳ Increasing, as outer loop, linearly increasing

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

↳ if $m \geq n$;
$m = 6$, $n = 1$

1/	↳ 6	1
2/	↳ 5	1
3/	↳ 4	1
4/	↳ 3	1
5/	↳ 2	1
6/	↳ 1	1

↳ takes $O(m)$ time to run (worst case)

↳ $n \geq m$
$n = 14$, $m = 2$

1/	↳ 14	2
2/	↳ 12	2
3/	↳ 10	2
4/	↳ 8	2
5/	↳ 6	2
6/	↳ 4	2
7/	↳ 2	2

↳ $O(\frac{14}{2} = 7)$
$O(\frac{n}{m})$ time to run
↳ $O(n)$ time

↳ Best case: $\Omega(1)$ → (m & n first equal)
↳ time complexity: $O(\max(m, n))$

↳ Decreasing pattern ~~as~~ each iteration reduces larger number so m or n's value decreases.

algo fun(n) Date _____ 20__

loop runs (n+1)

d) int i, j, k=0;
for(i=n/2; i<=n; i++){ ← n/2 → n times
for(j=2; j<=n; j=j*2){
k = k + n/2;
}
} return k;

↳ outer loop: should run (n+1) times
↳ will run for n/2 → n times
↳ ((n − n/2) + 1)
= n/2 iterations hence O(n)

↳ inner loop:

n = 16

	j (before)	j (after)	k (k += 8)
↳	2	4	8
↳	4	8	16
↳	8	16	32
↳	16	32	64

↳ iterations = 4, k increases by n/2 each iteration.

↳ $2^k \geq n$
$k \log_2 2 \geq \log n$
$k = \log n$
$O(\log n)$

↳ time complexity = $O(n \log n)$
↳ Increasing pattern, as outer loop increases from n/2 to n, & inner loop increases exponentially by 2^k.

Date ____ 20 ____

$k \geq n$

e) k=1;
for (i=0; i<n; i++) ← n times
{
for(j=0; j<n; j=j+k){ ← ~~n times~~
printf("%d \t", j);
}
k = k*2;
}

	i	j	k
1/	0	0	1
2/	:	1	1
3/	:	2	1
4/	:	:	:
5/	n=1	(n times) ~~inner loop~~	1

↳ i=1, k=2 (increament by 2)
↳ i=2, k=4 (increment by 2)

n=8
↳ iteration 1 (i=0, k=1)
↳ inner loop → runs 8 times (j=0,1,2,3,4,6,7)
↳ after loop → k=2
↳ iteration 2 (i=1, k=2)
↳ inner loop → runs 4 times (j=0,2,4,6)
↳ after loop → k=4
↳ iteration 3 (i=2, k=4)
↳ inner loop → j=0,4 → runs 2 times
↳ after loop → k=8
↳ iteration 4 (i=3, k=8)
↳ inner loop → j=0 → runs 1 times
↳ after loop → k=16

↳ $T(n) = n + \frac{n}{2} + \frac{n}{4} + \frac{n}{8} \cdots$ geometric series

$$\frac{a}{1-r} = \frac{n}{1-1/2}$$

$$= 2n$$

$T(k) < 2n$
$T(n) = 2n$
$O(2n) = O(n)$
↳ time complexity = $O(n)$
↳ pattern is increasing

Big O → $f(n) \leq c.g(n)$ for all $n \geq n_0$
Big Ω → $f(n) \geq c.g(n)$ for all $n \geq n_0$

Date ____ 20__

Q.5) a) Big O Proofs (Upper Bound)

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

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

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

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

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

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

↳ $c \geq 5$

↳ $c = 10$,

$5 - \frac{100}{n} + \frac{50}{n^2} \leq 10$

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

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

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

$0 \leq 5n^2 + 100n - 50$

$5n^2 + 100n - 50 \geq 0$

solved quadratically → $n \geq 0.48809$

hence $\boxed{n = 1}$ & $\boxed{c = 10}$

2) prove $\underbrace{n^2 + n\log n}_{f(n)} \in \underbrace{O(n^2)}_{c.g(n)}$

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

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

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

$c \geq 1$

↳ inequality ($\log n \leq n$) is true

↳ $\boxed{c = 2}$ & $\boxed{n \geq 1}$

↳ $c = 2$,

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

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

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

↳ $n = 1$: $\log(1) = 0 \leq 1$

↳ $n = 2$: $\log(2) = 0.69 \leq 2$

↳ $n = 3$: $\log(3) = 0.477 \leq 3$

Date _____ 20__

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

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

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

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

$$1 + \frac{1}{\log n} \leq c \quad \rightarrow \lim_{n\to\infty} 1 + \frac{1}{\log n} = 1 \leq c$$
$$= c \geq 1$$

↳ $c=2$,

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

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

$$\log n \geq 1$$

$n=1 \rightarrow \log(1) = 0 \geq 1$ ✗

$n=2 \rightarrow \log_2(2) = 1 \geq 1$ ✓

hence, $c \geq 1$, $n \geq 2$

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

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

$$n + 50 \leq c$$

↳ $(n+50)$ grows unbounded as $n \to \infty$

↳ $\boxed{n^4 + 50n^3 \notin O(n^3)}$ bec $\frac{n^4 + 50n^3}{n^3} = n + 50 \to \infty$,
so no constant 'c' can bound it

Date _____ 20__

b) Big Omega Proofs (Lower Bound)

5) prove $4n^2 - 1000n + 25 \in \Omega(n^2)$

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

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

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

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 4 \geq c$$

$$c \leq 4$$

↳ $c = 3$,

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

$$-\frac{1000n + 25}{n^2} \geq -1$$

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

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

$$n = 999.975 \quad n = 0.025$$

$$n \geq \lceil 999.975 \rceil = 1000$$

$n = 1000$

$c < 4$

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

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

$$1 + \frac{\log n}{n} \geq c \longrightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = 1 \geq c$$

$$c \leq 1$$

↳ $c = 1$,

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

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

$$\log n \geq 0$$

↳ $n = 1$, $\log_2(1) = 0 \geq 0$

↳ $n = 2$, $\log_2(2) = 1 \geq 0$

↳ $n = 3$, $\log_2(3) = 1.584 \geq 0$

↳ proved ⟶ $\boxed{n \geq 1}$ & $\boxed{c = 1}$

Date ____ 20__

7) prove $\log n \notin \Omega(n)$

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

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

↳ 'n' grows much faster than 'log n', for any constant [c > 0], [c . n] will be larger than log n, hence this is proved that $\log n \notin \Omega(n)$.

c) Big Theta Proofs (Tight Bound)

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

$$\boxed{c_1 g(n) \leq f(n) \leq c_2 g(n)}$$

↳ upper bound

$$\frac{f(n)}{g(n)} \leq c_2$$

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

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

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

$$c_2 \geq 10$$

↳ $c_2 = 11$,

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

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

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

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

$$n = \lceil 2.53 \rceil$$

$$n = 3$$

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

↳ lower bound

$$\frac{f(n)}{g(n)} \geq c_1$$

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

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 10 \geq c_1$$

$$c_1 \leq 10$$

↳ $c_1 = 9$,

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

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

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

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

$$n = \lceil 197.46 \rceil = 200$$

↳ lowerbound ⇒ $\boxed{c_1 = 9, n = 200}$ proved

Date ____ 20 ____

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

$c_1 g(n) \leq f(n) \leq c_2 g(n)$

↳ upperbound

$$\frac{n^2 + n\log n}{n^2} \leq c_2$$

$$1 + \frac{\log n}{n} \leq c_2$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1 \leq c_2$$

$$c_2 \geq 1$$

↳ $\boxed{c_2 \geq 1}$ & $\boxed{n \geq 1}$

↳ $c_2 = 2$,

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

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

$$\log n \leq n$$

↳ $n=1 \rightarrow \log_2(1) = 0 \leq 1$

↳ $n=2 \rightarrow \log_2(2) = 1 \leq 2$

↳ $n=3 \rightarrow \log_2(3) = 1.58 \leq 3$

↳ $n=4 \rightarrow \log_2(4) = 2 \leq 4$

↳ lowerbound

$$\frac{n^2 + n\log n}{n^2} \geq c_1$$

$$1 + \frac{\log n}{n} \geq c_1$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1 \geq c_1$$

$$c_1 \leq 1$$

↳ $\boxed{c_1 = 1}$ & $\boxed{n \geq 1}$

↳ $c_1 = 1$

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

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

$$\log n \geq 0$$

↳ $n=1 \rightarrow \log_2(1) = 0 \geq 0$

↳ $n=2 \rightarrow \log_2(2) = 1 \geq 0$

↳ $n=3 \rightarrow \log_2(3) = 1.58 \geq 0$

~~↳ $g(n) \quad n^2 + n\log n$~~

~~$(2)(1) \leq 1 + 1\log$~~

↳ $c_1 g(n) \leq n^2 + n\log n \leq c_2 g(n)$

$(1)(1)^2 \leq 1^2 + 1\log(1) \leq (2)(1)^2$

$1 \leq 1 \leq 2$ proved

QUALITY PAPER PRODUCTS

Date _____ 20___

10) prove $n \log n + 50 \in \theta(n \log n)$

$c_1 g(n) \leq f(n) \leq c_2 g(n)$

↳ upperbound

$$\frac{n \log n + 50}{n \log n} \leq c_2$$

$$\frac{n \log n}{n \log n} + \frac{50}{n \log n} \leq c_2$$

$$1 + \frac{50}{n \log n} \leq c_2$$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 1 \leq c_2$$

$$c_2 \geq 1$$

↳ $c_2 = 2$

$$1 + \frac{50}{n \log n} \leq 2$$

$$\frac{50}{n \log n} \leq 1$$

$$50 \leq n \log n$$

↳ $n = 5 \rightarrow 5 \log 5 = 50 \not\leq 11,6$

↳ $n = 10 \rightarrow 10 \log 10 = 33.2 \rightarrow 50 \not\leq 33.2$

↳ $n = 12 \rightarrow 12 \log 12 = 43.2 \rightarrow 50 \not\leq 43,01$

✓↳ $n = 14 \rightarrow 14 \log 14 = 53.2 \rightarrow 50 \leq 53.2$

↳ $\boxed{n \geq 14}$, $\boxed{c_2 \geq 1}$

↳ lowerbound

$$n \log n + 50 \geq c_1 g(n)$$

$$\frac{n \log n + 50}{n \log n} \geq c_1$$

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

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 1 \geq c_1$$

$$c_1 \leq 1$$

↳ $c_1 = 1$

$$1 + \frac{50}{n \log n} \geq 1$$

$c_1 = 1$, $n = 2$

↳ max (2, 10)