HOMEWORK 1

Name: ELSY FERNANDES

STUDENT ID: 1001602253

SECTION: SECTION 002 (MW 2:30-3.50PM)

Q.1) a) closed form, dominant term, ⊖ not required

```
for (i=1; i<=N; i=i+1)
for (k=1; k<=i; k=2*k)
for (t=1; t<=N; t=2*t)
  printf ("c");
```

iter	i	'k' values	for-k iteration count	't' values	for 't' iteration count	Total iteration from for-k for-t loop
1.	1	1	1	1,2,4...2^P where, 2^P is less than Equal to N	lgN	(1) lgN
2.	2	1,2	2	1,2,4...2^P where, 2^P is less than Equal to N	lgN	(2)lgN
3.	3.	1,2	2	1,2,4...2^P where, 2^P is less than Equal to N	lgN	(2)lgN
4	4	1,2,4	3	1,2,4...2^P where, 2^P is less than equal to N	lgN	(3)lgN
⋮	⋮	⋮				

itr	i	'k' Values	iteration Count for 'k'	't' Values	iteration Count for 't'	Total iteration from for k & for t loop
⋮						
N-2	N-2	1, 2, 4 ..... N-2	$\lg(N-2)+1$ (approximately)	1, 2, 4 ... 2^P where, 2^P is less than equal to N	$\lg N$	$(\lg(N-2)+1)\lg N$
N-1	N-1	1, 2, 4 .... N-1	$\lg(N-1)+1$ (approximately)	1, 2, 4 ... 2^P where, 2^P is less than equal to N	$\lg N$	$(\lg(N-1)+1)\lg N$
N	N	1, 2, 4 .... N	$\lg N+1$ (approximately)	1, 2, 4 .. 2^P where, 2^P is less than equal to N	$\lg N$	$(\lg N+1)\lg N$

<u>Summation</u>: $\lg N + 2\lg N + 2\lg N + 3\lg N + 3\lg N + 3\lg N + 3\lg N$

$+ 4\lg N + 4\lg N + 4\lg N + \cdots$

$\cdots + (\lg(N-2)+1)\lg N + (\lg(N-1)+1)\lg N$

$+ (\lg(N)+1)\lg N$

Note: $\lg(N-2)+1$ is a approximate value it can be ~~$\lg(N-2)$~~ $\lg(N-2)$

Same goes for $\lg(N-1)+1$ $\lg N+1$

Q.1) b) closed form, dominant term, Θ required ②

```
for (i=1; i<=N ; i=i+1)
for (k=1; k<=S ; k++)
for (t=1; t<=i ; t++)
printf ("D");
```

itr	i	'k' Values	Iteration Count for 'k'	't' Values	Iteration Count for 't'	Total iteration from loop k for t loop
1	1	1, 2, 3, ..., S	(S)	→ 1 → 1 → 1 → 1	(1)	(1) S
2	2	1, 2, 3, ..., S	(S)	→ 1, 2 → [illegible]	(2)	(2) S
3	3	1, 2, 3, ..., S	(S)	1, 2, 3	(3)	(3) S
⋮	⋮	⋮	⋮	⋮	⋮	⋮
N-1	N-1	1, 2, 3, ..., S	(S)	1, 2, 3 ... (N-1)	(N-1)	(N-1) S
N	N	1, 2, ...	(S)	1, 2, 3 ... N	N	(N) S

Summation : $S + 2S + 3S + 4S + \ldots (N-1)S + NS$

$S(1 + 2 + 3 + \ldots (N-1) + N)$

$S\left(\frac{N(N+1)}{2}\right)$

Closed form of solution : $S\left(\frac{N(N+1)}{2}\right)$

Simplify : $\Rightarrow \frac{SN^2}{2} + \frac{SN}{2}.$

Dominant term : SN^2

$(N) = \Theta(N^2)$

Q.1) e) closed form, dominant term, Θ required


```
for(i=1; i<=N; i=i+1) {
 for(k=1; k<=N; k++)
  for (t=1; t<=k ;t++)
   printf ("E");
 for (k=1; k<=M; k++)
 for (t=1; t<=k; t++)
   printf ("F");
```

itr	i	'k' value	iteration Count for k	't' value	iteration Count for 't'	Total iteration from loop k for loop t	
1.	1	1, 2, ... → N	N	1, 2, 3 .... N	$\frac{N(N+1)}{2}$	$\frac{N(N+1)}{2}$	$\frac{N(N+1)}{2} + 1$
	1	1	1	1	1	1	
2.	2	1, 2, 3 ... → N	N	1, 2, 3 ---→ N	$\frac{N(N+1)}{2}$	$\frac{N(N+1)}{2}$	$\frac{N(N+1)}{2} + 2$
	2	2	1	1, 2	2	2	

itr	i	'k' Value	iteration count for 'k'	't' Value	iteration Count for 't'	Total iteration from loop 'k' for loop 't'	
3	3	1, 2, 3 ... N	N	1,2,3 ... N	$\frac{N(N+1)}{2}$	$\frac{N(N+1)}{2}$	$\frac{N(N+1)}{2} + 3$
	3	3	1	1,2,3	3	3	
N−1	N−1	1, 2 ... N	N	1,2,3 ... N	$\frac{N(N+1)}{2}$	$\frac{N(N+1)}{2}$	$\frac{N(N+1)}{2} + M-1$
	N−1	N−1	1	1,2,3 ... M−1	M−1	M−1	
N	N	1, 2, 3 ... N	N	1,2,3 ... N	$\frac{N(N+1)}{2}$	$\frac{N(N+1)}{2}$	$\frac{N(N+1)}{2} + M$
	N	M	1	1,2,3 ... M	M	M	

Summation from the table:

$$= \frac{N(N+1)}{2} + 1 + \frac{N(N+1)}{2} + 2 + \frac{N(N+1)}{2} + 3$$

$$+ \cdots \frac{N(N+1)}{2} + M-1 + \frac{N(N+1)}{2} + M$$

Seperate 'N' and 'M' terms;

$$= \frac{N(N+1)}{2} + \frac{N(N+1)}{2} + \cdots \frac{N(N+1)}{2}$$

$$+ 1 + 2 + 3 + \cdots M-1 + M$$

[? in AP

$T_n = a + (n-1)d$

$n = \frac{T_n - a}{d} + 1$

$= \frac{N-1}{1} + 1$

$= N$]

$$= \left(\frac{N(N+1)}{2}\right) N + \frac{M(M+1)}{2}$$

$$= \left(\frac{N^2+N}{2}\right) N + \frac{M^2+M}{2}.$$

$$= \frac{N^3+N^2}{2} + \frac{M^2+M}{2}. \Rightarrow \frac{N^3+N^2+M^2+M}{2}$$

closed form: $\frac{N^3+N^2+M^2+M}{2}$

dominant term: $N^3 + M^2$.

$$T(M,N) = \Theta(N^3+M^2)$$

Q)2. How many times the loop iterate

```
for(i=1; i<=N; i=i+7);
```

There is no Body of loop. Hence $T(N) = 0$

Q)3. Give the theta time complexity and show your work in the best way you can.

```
i=0;
while (i <= N) {
for(t=0, k=1; k<N; t=t+1, k=2*k)
  printf("G");
  i=i+t;
}
```

Say Suppose $\boxed{N=16}$. My calculations goes as shown in the table Below.

itr	i	't' values	't' count	'k' value	'k' count	Total iteration of k for this i
1	0	In case of N=16 0, 1, 2, 3, 4 [Here last term is $\lg_2 16 = 4$]		In case N=16 1, 2, 4, 8, 16		lgN
		In general, 0, 1, 2, 3 ... lgN	lgN+1	In general, 1, 2, 4 .. 8 ... 2^P 2^P is less than N	lgN	lgN

itr	i	't' values	't' count	'k' value	'k' count	Total Iteration of k for this i
2	when N=16 i=i+t i=0+lgN i=lgN when N=16 lgN=4 i=4	when N=16 0,1,2,3,4 [last term is=4 lg 16 = 4]		when N=16 1,2,4,8,16		lgN
	In general i=0+lgN lgN	In general, 0,1,2...lgN	lgN+1	In general, 1,2,4...8 ...2P 2P les than N	lgN	lgN
3.	N=16 i=i+t i=4+4 i=8	when N=16 0,1,2,3,4 [last term is 4 lg16=4		when N=16 1,2,4,8,16		
	In general i=lgN+lgN =2lgN	In general, 0,1,2...lgN	lgN+1	In general, 1,2,4...8 .. 2P 2P less than N	lgN	lgN
	⋮	⋮	⋮	⋮	⋮	⋮
4	i=NlgN less than N	In general, 0,1,2,...lgN	lgN+1	In general, 1,2,4...2P 2P less than N	lgN	lgN

→ value of i is in Arithmatic progression

→ i takes values from $[0, \lg N, 2\lg N \dots N\lg N]$

where $N\lg N$ is less than N

→ So, total count of loop k for this i

can be calculated as:

$$[\text{A.P of } i]\,[\lg N + \lg N + \dots \lg N] \quad \text{—} (1)$$

To find Number of terms in AP

$T_n = a + (n-1)d$ $\qquad$ $T_n \rightarrow$ last term

$a \rightarrow$ First term

$$n = \frac{T_n - a}{d} + 1$$

$n \rightarrow$ no of terms

$d \rightarrow$ difference

$$n = \frac{(N-1) - 0}{\lg N} + 1$$

$T_n = (N-1)$ approximately

$d = 2\lg N - \lg N = \lg N$

Substitute in (1)

$$= \frac{(N-1) - 0 + \lg N}{\lg N}$$

$$= \left(\frac{N-1 + \lg N}{\lg N}\right) \lg N$$

$$= \underline{\underline{N-1+\lg N}}$$

$$\underline{\underline{T(N) = \Theta(N)}}$$

Q) 4) write the code that has time complexity Below and after that show why it has that complexity

$$T(n) = 1 + 3^1 + 3^2 + \dots + 3^n$$

```
for(i=0; i<=N; i++)
for (k=0 ; k<pow(3,i) ; k=k+1)
  printf ("3");
```

itr	i	'k' Values	Iteration Count for 'k'	Total iteration Count for K
1	0	0 [less than 3^0]	(1)	1
2	1	0 [less than 3^1] 1 [less than 3^1] 2 [less than 3^1]	(3)	3^1
3	2	0 [less than 3^2] 1 [less than 3^2] 2 [less than 3^2] 3 [less than 3^2] 4 [less than 3^2] 5 [less than 3^2] 6 [less than 3^2] 7 [less than 3^2] 8 [less than 3^2]	(9)	3^2

itr	i	'k' values	iteration count for 'k'	Total iteration count for 'k'
4	3	0 [less than 3^3] 1 [less than 3^3] 2 [less than 3^3] ⋮ 26 [less than 3^3]	27	3^3
N−1	N−1	0 [less than 3^{n-1}] 1 2 ⋮	3^{N-1}	3^{N-1}
N	N	0 [less than 3^n] 1 2 ⋮	3^N	3^N

$$T(N) = 1 + 3 + 3^2 + \cdots + 3^{N-1} + 3^N$$

Q. 5) Find the Dominant terms and write Θ for function Below.

$N^3 + 500N^2 + NM + 10^6$

Dominant term $= N^3 + NM$

$T(N, M) = \Theta(N^3 + NM)$

$100N^3 + 20N^2 + 15M + 5N$

Dominant term $= 100N^3 + 15M$

$T(N, M) = \Theta(N^3 + M)$

Q.6) (a)

```
//run for N=10, N=100, N=300, N=1000

void runtime_increment (int N){
   int i, k,t, res=0;
  for (i= 1 ; i <=N; i=i+1)
  for (k=1; k<=i; k++ )
  for (t=1; t<=N; t++)
    res = res + 1 ;
```

N	Result (in Sec/ms)	Result
10	20ms	It ran in less than a Second
100	30ms	It ran in less than a Second
300	50ms	It ran in less than a second
1000	1s	It took almost 1 Second to run.

// run for N=10, N=100, N=300, N=1000

```
Void runtime_print (int N){
 int i, k, t;
for (i=1; i<=N; i=i+1)
for (k=1; k<=i; k=k++)
for (t=1; t<=N; t++)
 printf ("A");
```

N	Result (in sec/ms)	Result
10	10ms	It ran in less than a second
100	1s	It ran in one second
300	30s	It ran for half a minute
1000	1159 sec	It ran for 19m 9second

```
// run for N=10, N=15, N=20, N=25, N=30, N=100, N=1,000
void runtime_pow(int n) {
  int i, res=0;
  for (i=1; i<= pow(2.0, (double) N); i=i+1)
    res = res + 1;
```

N	Result (s|ms)	Result
10	20ms	Ran in less than a second
15	20ms	Ran in less than a second
20	30ms	Ran in less than a second
25	400ms	Ran in less than a second
30	12s	Ran in More than 10s.

- the runtime Effect of Replacing the res = res+1 with 'print' Instruction for runtime_increment and runtime-print

Runtime_Effect

N	Runtime_increment	Runtime_print
300	50ms [Ran in less than second]	30s [Ran in Half a minute]
1000	1s [It took 1 second to run]	1159 [It took 19m to run]

→ How the Runtime depends on the Value of N for runtime-increment ?

As show in the table for runtime_increment

For N = 10 → Runtime = 20ms

N = 100 → Runtime = 30ms

N = 300 → Runtime = 50ms

N = 1000 → Runtime = 1s

Runtime Increases when value of N increases

→ How Much Faster ther runtime-Pow becomes too Slow

Since, the code has Exponential power

Runtime depends Exponentially on N.

Q.6) b) which of the three Functions above (runtime_increment runtime_print and runtime_pow) has the time complexity (closer) (or more similar) to that of Runtime_rec ?

N	Result (s | ms)	Result
10	3s	Runs in 3s
15	4s	Run in 4s
20	11s	Runs in 11s
25	12s	Runs in 12s
30	18s	Run in 18s
100	8m 8s	Terminated after 8m 8s.

→ Runtime_Rec grows Exponentially

→ It is similar to runtime_pow