Institute of Business & Information Technology

University of the Punjab

Quaid-e-Azam Campus, Lahore

Sheet No. _____

Assignment 01

Spring Term 2018

Code IT-466
Title Analysis of Algorithm
Due Date: Feb 21, 2018 before 1500hr

Degree BBIT
Batch F15-IT Specialization
Marks 100

Instructions:

1. Do not forget to pray before starting to attempt the paper. Trust me it helps.
   Remember! SOMEONE is always with you (Be Relaxed), and HE is also watching you (Be Honest)
2. **Question Paper is SELF EXPLANATORY. Understanding the Question Paper is part of Solution.**
3. Nothing Beyond the Finish Line will be Evaluated. Back Side of Pages is Beyond Finish Line.
4. For Calculations etc. Use the back side of the pages.
5. Error in Question will be advantageous to Student.
6. Read the Questions carefully before attempting.
7. Solve your paper using Black/Blue Pen only.
8. Attempt All Questions in a Precise Fashion.
9. Switch Off your Cellular Phones.
10. Manage Your Time.

Solution

GOOD LUCK

Reg. No: _____ **Name:** _____

Class: IT Specialization **Date:** Friday, Feb 16, 2018

Section: _____ **Signature:** _____

Q 01.	Q 02.	Q 03.	Q 04.	Q 05.	Q 06.	Q 07.	Q 08.	Q 09.	Q 10.	Total
20	20	40	20							100

Assistant's Signature

Examiner's Signature

Do Not Open until YOU are Advised by the Invigilator.

Question No 01. 20

Find the Time Complexity of the Following Iterative Algorithms, Show Working. 3

```
int x=1, y=1;
while ( y <= n ){
    y = x * x;
    Print(y );
    x++;
}
```

$x = 1, 2, 3 \ldots k$

$y = 1^2 \quad 2^2, 3^2 \ldots k^2$

$k^2 <= n$

$\Rightarrow k <= \sqrt{n}$

So $T(n) = O(\sqrt{n})$

4

```
for(i=1; i <= n/2 ; i++)
    for(int j=1; j <= i³ ; j++)
        for(int k=1; k <= n; k=k * 2)
            Print("IBIT-PU");
```

$i = 1$	2	3	$n/2$
$j = 1^3$ times	$j = 2^3$ time	$j = 3^3$ times	$(n/2)^3$ times
$k = \log_2 n$	$k = \log_2 n$	$k = \log_2 n$	
$1 \times \log_2 n$	$2^3 \log_2 n$	$3^3 \log_2 n$	$\left(\frac{n}{2}\right)^3 \log_2 n$

$T(n) = 1^3 \log_2 n + 2^3 \log_2 n + 3^3 \log_2 n \ldots + \left(\frac{n}{2}\right)^3 \log_2 n$

$= \log_2 n \sum_{i=1}^{n/2} i^3 = \log_2 n * \left[\frac{\frac{n}{2}\left(\frac{n}{2}+1\right)}{2}\right]^2$

$\Theta(n^4 \log_2 n)$

3

```
for (i = 1; i <= n; i += c)
    for (j = i+1; j <= n; j *= c)
        Print("IBIT");
```

$i = 1 \ldots n/c$

$j = \log \frac{n+1}{c}$

$T(n) = \frac{n}{c} \log\left(\frac{n+c}{c}\right) = O(n \log n)$

4

```
for(j=1; j <= n; j++)
    for(i=1; i <= j^2; i++)
        for(c=1; c <= n/2; c++)
            print("IBIT");
```

j — n time
i — $\sqrt{n}$ times
c — $\frac{n}{2}$ time.

$$T(n) = O\left(n \times \sqrt{n} \times \frac{n}{2}\right) = O(n^{5/2})$$

3

```
for (i = n/2; i > 0; i -= c)
    for (j = i+1; j <=n; j += c)
        Print("IBIT");
```

i — $n/2c$ times
j — $\frac{n-1}{c}$ time

$$T(n) = \frac{n}{2c} \times \frac{n-1}{c} = \frac{n(n-1)}{c^2}$$

$$= O(n^2)$$

4

```
for(int i=1;i <= n;i=i*2)
    for(int j=1; j <= 2*(i-1); j++)
        Print("IBIT-PU")
    for(int k=i; k <= (10-i); k++)
        Print("AoA")
```

i	1	2	4	8	16	... $\log_2 n$
j	0 time	2 times	6	14	30	
k	1 — 10 time	2 — 8 times	4 → 6	0	0	

$i = \log_2^n$

$j = 2 \log$

$$1 + \log_2^n \times 2(\log_2^n - 1)$$

$$= \log_2^n \times (2\log_2^n - 2 + 1)$$

$$= \log_2^n \times (n \log_2^n - n)$$

$$O(n \log_2^2 n - n \log_2^n)$$

$$= O(n \log_2^2 n)$$

Question No 02. 20

Find the Time Complexity of the Following Recursive Algorithms, Show Working. 5

```
void Func(int n){
    if (n==1)
        return;
    for (i=1;i <= n; i=i*5)
        Print("IBIT-PU");
    Func(n-1);
}
```

$T(n) = T(n-1) + \log_5 n$

$T(n) = T(n-2) + \log_5 n-1 + \log_5 n = T(n-2) + \log_5 n(n-1)$

$T(n) = T(n-3) + \log_5 n(n-1)(n-2)$

$T(n) = T(n-4) + \log_5 n(n-1)(n-2)(n-3)$

$T(n-k) = T(n-k-1) + \log_5 n-k$

$n-k-1 = 1 \Rightarrow k = n-2$

$T(n) = T(1) + \log_5 n(n-1)(n-2) \ldots (2) = T(1) + \log_5 n!$

$= 1 + O(n \log_5 n) \Rightarrow O(n \log_5 n)$

```
int findMax(int[] a, int size){
    int Max;
    if(size == 1){
        return a[0];
    }else{
        Max = findMax(a, size - 1);
        if(Max > a[size - 1])
            return Max;
        else
            return a[size - 1];
    }
}
```

4

$T(S) = T(S-1) + 1$

$T(S) = T(S-2) + 1+1$

$T(S) = T(S-3) + 1+1+1$

$T(S) = T(S-4) + 1+1+1+1$

$T(S) = T(S-k) + (1+1+1+1+\ldots k \text{ times})$

$S-k = 1$

$\Rightarrow k = S-1$

$T(S) = T(0) + (1+1+1+1 \ldots S \text{ times})$

$= 1+1+1+\ldots\ldots + (S-1) \text{ times}$

$= S$

$T(S) = O(S)$

4

```
int Fib(int n) {
   if(n == 1 OR n == 2)
      return 1;
   else
      return(Fib(n - 1) + Fib(n - 2));
}
```

$T(n) = T(n-1) + T(n-2) + c$

$T(n-1) \approx T(n-2)$

$T(n) = 2T(n-1) + c$

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

$T(n) = 2^3 T(n-3) + 7c$

$T(n) = 2^4 T(n-4) + 15c$

$T(n) = 2^k T(n-k) + (2^k - 1)c$

$n - k = 0 \Rightarrow n = k$

$T(n) = 2^n T(0) + (2^n - 1)c$

$= 2^n (1+c) - c$

$= O(2^n)$

Upper Bound

$T(n-2) \approx T(n-1)$

$T(n) = 2T(n-2) + c$

$T(n) = 2^2 T(n-4) + 3c$

$T(n) = 2^3 T(n-6) + 7c$

$T(n) = 2^k T(n-2k) + (2^k - 1)c$

$n - 2k = 0 \Rightarrow k = n/2$

$T(n) = 2^{n/2} T(0) + (2^{n/2} - 1)c$

$= 2^{n/2}(1+c) - c$

$= O(2^{n/2})$

Lower bound

$T(n) = 3T(n-2) + n^3$ 7

$T(n) = 3T(n-2) + n^3$

$T(n) = 3^2 T(n-4) + 3(n-2)^3 + n^3$

$T(n) = 3^3 T(n-6) + 3^2(n-4)^3 + 3(n-2)^3 + n^3$

$T(n) = 3^4 T(n-8) + 3^3(n-6)^3 + 3^2(n-4)^3 + 3(n-2)^3 + n^3$

$T(n) = 3^4 T(n-4.2) + 3^3(n-6)^3 + 3^3(n-4)^3 + 3(n-2)^3 + n^3$

$T(n) = 3^k T(n-k.2) + 3^{k-1}(n-(k-1)2)^3 + \cdots + 3^2(n-4)^3 + 3^1(n-2)^3 + 3^0 n^3$

$n - 2k = 1 \Rightarrow k = (n-1)/2$

$T(n) = 3^{(n-1)/2} T(1) + 3^{(n-1)/2 - 1}\left[n - \left(\frac{n-1}{\cancel{2}}\right)\cancel{2}\right]^3 + \cdots + 3^2(n-4)^3 + 3^1(n-2)^3 + 3^0 n^3$

$\frac{3^n}{\sqrt{3}} \cdot 1 + 3^{(n-1)/2}(1)^3 + \cdots + 3^2(n-4)^3 + 3(n-2)^3 + n^3$

$= \frac{3^n}{\sqrt{3}} + \sum_{i=1}^{(n-1)/2} 3^i \cdot 2^3$

$\leq \frac{3^n}{\sqrt{3}} + \frac{n^2(n+1)^2}{4}$

$\leq \frac{3^n}{\sqrt{3}} + \frac{n^4}{4} \Rightarrow T(n) = O(3^n + n^4)$

40

Question No 03.

Find the Time Complexity of the Following Recursive Algorithms using Recursion Tree, Show Working. 8

$T(n) = 5T\left(\frac{n}{5}\right) + cn^5$

	Level	Node	WD
cn^5	0	1	cn^5
$c\left(\frac{n}{5}\right)^5$	1	5	$5c\left(\frac{n}{5}\right)^5$
$c\left(\frac{n}{5^2}\right)^5$	2	5^2	$5^2 c\left(\frac{n}{5^2}\right)^5$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
i	i	5^i	$5^i c\left(\frac{n}{5^i}\right)^5$
			$5^{h-1} \cdot c\left(n/5^{h-1}\right)^5$
$T(1)$	h	5^h	$5^h T(1)$

$h = \log_5 n$

$T(n) = 5^h T(1) + \sum_{i=0}^{h-1} c\left(\frac{n}{5^i}\right)^5 \cdot 5^i$

$= n + \sum_{i=0}^{\log_5 n - 1} c \cdot 5^{\log_5 n - 1} \cdot \frac{n^5}{5}$

$= n + c\sum_{i=0}^{\log_5 n - 1} 5^i \cdot \frac{n^5}{5^{4i}}$

$= n + cn^5 \sum_{i=0}^{\log_5 n - 1} \frac{1}{5^{4i}}$

$= n + \frac{cn^5}{5^4} \sum_{i=0}^{\log_5 n - 1} \frac{1}{5^i}$

$= n + \frac{cn^5}{5^4} \frac{1 - 1/5^{\log_5 n}}{1 - 1/5}$

$= n + \frac{c}{5^4} n^5 \frac{1 - \frac{1}{n}}{5/4}$

$= n + Qn^5 \Rightarrow O(n^5)$.

$T(n) = 3T\left(\frac{n}{2}\right) + c\sqrt{n}$ 8

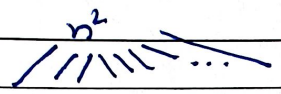
	Level	Nodes	Work Done
$T(n)$	0		
$T(n/2)\ T(n/2)\ T(n/2)$	1		
$c\sqrt{n}$	0	1	$c\sqrt{n}$
$c\sqrt{n/2}\ c\sqrt{n/2}\ c\sqrt{n/2}$	1	3^1	$3c\sqrt{\frac{n}{2}}$
$c\sqrt{\frac{n}{2^2}}\ c\sqrt{\frac{n}{2^2}}\ c\sqrt{\frac{n}{2^2}}$	2	3^2	$3^2 c\sqrt{\frac{n}{2^2}}$
$\vdots$			
i $\quad c\sqrt{\frac{n}{2^i}}$	i	3^i	$3^i c\sqrt{\frac{n}{2^i}}$
$\vdots$			
			$3^{h-1} c\sqrt{n/2^{h-1}}$
$T(1)$	h	3^h	$3^h T(1)$

$h = \log_3 n$ So $T(n) = 3^h T(1) + \sum_{i=0}^{h-1} c\sqrt{\frac{n}{2^i}} \cdot 3^i$

$= n + c\sqrt{n} \sum_{i=0}^{\log_3 n - 1} \left(\frac{3}{2}\right)^i = O(n)$

7

$T(n) = 16T\left(\frac{n}{4}\right) + n^2$



Level	Nodes	WD
0	1	n^2
1	4^2	$4^2 \cdot \frac{n^2}{4^2} = n^2$
2	4^4	$4^4 \cdot \left(\frac{n}{4^2}\right)^2 = n^2$
i	4^{2i}	$4^{2i} \cdot \left(\frac{n}{4^i}\right)^2 = n^2$
h	4^h	$4^{2(h-1)}\left(n/4^{h-1}\right)^2 = n^2$ $4^h \cdot T(1)$

$T(1)$

$$T(n) = 4^h T(1) + \sum_{i=0}^{h-1} n^2$$

$$= n + n^2 \sum_{i=0}^{\log n - 1} 1$$

$$= n + n^2 \log n \Rightarrow T(n) = O(n^2 \log n).$$

7

$T(n) = 5T\left(\frac{n}{5}\right) + \frac{n}{\lg n}$ Build Tree Yourself

Level	Nodes	WD
0	1	$n/\log n$
1	5^1	$n/\log n/5$
2	5^2	$n/\log n/5^2$
i	5^i	$n/\log n/5^i$
$h-1$	5^{h-1}	$n/\log n/5^{h-1}$
h	5^h	$5^h T(1)$

$h = \log_5 n$

$$T(n) = 5^h T(1) + \frac{n}{\log n} + \frac{n}{\log n/5} + \frac{n}{\log n/5^2} \cdots \frac{n}{\log n/5^{h-1}}$$

$$= 5^h T(1) + \sum_{i=0}^{h-1} \frac{n}{\log n/5^i}$$

$$= n + n \sum_{i=0}^{\log_5 n - 1} \frac{1}{\log \frac{n}{5^i}} = n + n\left[\frac{\frac{1}{\log n/5}\left(1 - \left(\frac{1}{\log n/5}\right)^{h-1}\right)}{1 - \frac{1}{\log n/5}}\right]$$

OR

OR

$$T(n) = n + n\left[\frac{\frac{1}{\log n/5}\left(1 - \frac{1}{(\log n/5)^{\log_5 n}}\right)}{(\log n/5 - 1)/\log n/5}\right]$$

$$= n + n\left[\frac{(\log n/5)^{\log_5 n} - 1}{(\log n/5)^{\log_5 n}(\log n - \log 5 - 1)}\right]$$

$$= n + n\left[\frac{(\log n)^{\log_5 n} - 5^{\log_5 n} - 1}{(\log n)^{\log_5 n} - \log_5 n(\log n - 1)}\right]$$

$$= n + n(\log n)^{\log_5 n} = O(n + n(\log n)^{\log_5 n})$$

10

$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{4}\right) + 3n^2 + 2n$$

Question No 04. **25**

Write down the Formal Statement of Master Theorem 5

$T(n) = aT(\frac{n}{b}) + b^k \log^p n$

$a >= 1, b > 1, k >= 0 \quad p \in R$

i. $a > b^k \Rightarrow T(n) = O(n^{\log_b a})$

ii $a = b^k$

1. $p > -1 \Rightarrow T(n) = O(n^{\log_b a} \log^{p+1} n)$
2. $p = -1 \Rightarrow T(n) = O(n^{\log_b a} \log\log n)$
3. $p < -1 \Rightarrow T(n) = O(n^{\log_b a})$

iii $a \cancel{>} b^k$

$p >= 0 \Rightarrow T(n) = O(n^k \log^p n)$

$p < 0 \Rightarrow T(n) = O(n^k)$

Find the Time Complexity of the Following Recursive Algorithms using Recursion Tree, Show Working.

$T(n) = 4T\left(\frac{n}{2}\right) + n^2\sqrt{n}$ 3

$a = 4, \quad b = 2, \quad k = 3/2, \quad p = 0$

$a > b^k$

$T(n) = O(n \log_b^a) = O(n^{\log_2 4}) = O(n^2)$

$T(n) = 2T\left(\frac{n}{2} + 17\right) + n$ 3

$a = 2, \quad b = 2, \quad k = 1, \quad p = 0$

$a = b^k$

and $p > -1 \quad T(n) = O(n^{\log_b a} \log^{p+1} n)$

$= O(n^{\log_2 2} \log^1 n) = O(n \log n)$

$T(n) = 7T\left(\frac{n}{2}\right) + \Theta(n^2 \log^3 n)$ 3

$a = 7, \quad b = 2, \quad k = 2, \quad p = 3$

$a > b^k$

$T(n) = O(n^{\log_b a}) \Rightarrow T(n) = O(n^{\log_2 7})$

$\approx O(n^3)$

Time Complexity of Recursive Functions by Master Theorem

$T(n) = 2T\left(\frac{n}{2}\right) + n^2 \lg n$ 3

$a = 2, \quad b = 2, \quad k = 2, \quad p = 1$

$a < b^k$

$p > 0, \quad T(n) = O(n^k \log^p n)$

$\Rightarrow \quad T(n) = O(n^2 \log n)$

$T(n) = 4T\left(\frac{n}{2}\right) + n^2 lgn$ 3

$a = 4, \quad b = 2, \quad k = 2, \quad p = 1$

$a = b^k$

$p > -1 \Rightarrow T(n) = O(n^{\log_b a} \log^{p+1} n)$

$\Rightarrow T(n) = O(n^{\log_2 4} \log^{1+1} n)$

$= O(n^2 \log^2 n)$

Finish Line

Nothing beyond this line will be evaluated