Institute of Business & Information Technology

University of the Punjab

Quaid-e-Azam Campus, Lahore

Sheet No. _____

Solution

Assignment 01

Spring Term 2017

Code IT-
Title Analysis of Algorithm
Due Date: Mar 27, 2017 before 1100hr

Degree BBIT
Batch F14-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.

GOOD LUCK

Reg. No: _____ Name: _____

Class: IT Specialization Date: Tuesday, March 21, 2017

Section: _____ Signature: _____

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

Assistant's Signature

Examiner's Signature

Do Not Open until YOU are Advised by the Invigilator.

10

Question No 01.

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

```
for(i=2 ; i <=n/2 ; ++i){ n/2
    if(n%i==0){ n/2 - 1
        flag=1; n/2 - 1
        break; n/2 - 1
    }
}
```

$T(n) = \frac{n}{2} + \frac{n}{2} - 1 + \frac{n}{2} - 1 + \frac{n}{2} - 1$

$T(n) = 4\frac{n}{2} - 3$

$T(n) = O(n)$

```
for(i=0 ; i <=n/2 ; i++){ n/2
    for(int j=0 ; j <=n/3 ; j++){ n/3 (n/2 - 1)
        for(int k=0 ; k<=n ; k++){ n (n/3 - 1)(n/2 - 1)
            c[i,j,k]= a[i,j,k] * b[j,i,k]; (n-1)(n/3 - 1)(n/2 - 1)
        }
    }
}
```

$T(n) = n/2 + n/3\,(n/2 - 1) + n(n/3 - 1)(n/2 - 1) + (n-1)(n/3 - 1)(n/2 - 1)$; $T(n) = n/2 + n/3\,(n/2 - 1) + n(n/3 - 1)(n/2 - 1) + (n-1)(n/3 - 1)(n/2 - 1)$; $T(n) = n/2 + n^2/6 - n/3 + (n^2/3 - n)(n/2 - 1) + (n^2/3 + n - n/3 + 1)(n/2 - 1)$

$T(n) = n/2 + \cancel{n^2/6} - \cancel{n/3} + n^3/6 - n^2/3 - \cancel{n^2/2} + \cancel{n} + n^3/6 + \cancel{n^2/2} - n^2/6 + n/2 - \cancel{n^2/3} - \cancel{n} + \cancel{n/3} - 1$

$T(n) = n/2 + n^3/6 - n^2/3 + n^3/6 + n/2 - n^2/3 - 1$

$T(n) = 2n/2 + 2n^3/6 - 2n^2/3 - 1$

$T(n) = O(n^3)$

```
while (n >= 1) { — log n
    n = Math.Log(n,base); — log n - 1
    count++; — log n - 1
}
```

Suppose $n = 1000$; 1000 500 250 150 ---- 10
3 2.7 2.14 2 1

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

$T(n) = O(\log n)$ Iterative logarithm, Basically in Computer science the iterative logarithm of n written as $\log^* n$ (usually read as "log star") is the number of times the algorithm function must be iteratively applied before the result is less than or equal to one.

```
for (c = 0; c < m ; c++){ m
    for (d = 0; d < q; d++){ q(m-1)
        for (k = 0 ; k < p ; k++){ p(m-1)(q-1)
            sum = sum + first[c][k] * second[k][d]; (m-1)(q-1)(p-1)
        }
        multiply[c][d] = sum; (m-1)(q-1)(p-1)
        sum = 0; (m-1)(q-1)(p-1)
    }
}
```

Assume $m = q = p = n$

Then $T(n) = n + n(n-1) + n(n-1)(n-1) + (n-1)(n-1)(n-1) + (n-1)(n-1)$
$(n-1) + (n-1)(n-1)(n-1) + (n-1)(n-1)(n-1)$ $\because (a-b)^3 = (a-b)(a^2+ab+b^2)$

$T(n) = n + n^2 - n + (n^2-n)(n-1) + (n-1)(n^2+n+1) + (n-1)(n^2+n+1)$
$+ (n-1)(n^2+n+1)$; $T(n) = \cancel{n} + \cancel{n^2} - \cancel{n} + n^3 - \cancel{n^2} - \cancel{n^2} + \cancel{n} + n^3 + \cancel{n^2}$
$+ \cancel{n} - \cancel{n^2} - \cancel{n} - 1 + n^3 + \cancel{n^2} + n - \cancel{n^2} - \cancel{n} - 1 + n^3 + \cancel{n^2} + \cancel{n} - \cancel{n^2} - \cancel{n} - 1$

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

$T(n) = O(n^3)$

```
for ( i = 1; i <=n; i += c) { n
    for (int j = 1; j <=n; j += c) { n(n-1)
        Print("IBIT"); (n-1)(n-1)
    }
}
```

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

$T(n) = \cancel{n} + n^2 - \cancel{1} + n^2 - \cancel{n} - n + \cancel{1}$

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

$T(n) = O(n^2)$

Whenever the constant is added to step variable it completes $O(n)$ time complexity, so here nested loops compute $O(n^2)$ time complexity.

```
for (i = n ; i > 0 ; i += c){ ⟶ n
    for (j = i+1 ; j <=n ; j += c) { ⟶ for (j = i+1 ; j>0 ; j-=c) ⟶ n(n-1)
        Print("IBIT"); (n-1)(n-1)
    }
}
```

Correct:

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

$= \cancel{n} + n^2 - \cancel{1} + n^2 - \cancel{n} - n + 1$

$= 2n^2 - n$

$T(n) = O(n^2)$

Whenever a constant is added to the loop, step variable it computes $O(n)$ time complexity, so here, nested loops compute $O(n^2)$ time complexity.

```
for (i = 1 ; i <=n ; i *= c) {
   Print("IBIT");
}
```

— $\log n$

$\log n - 1$

Suppose $n=6$; $c=2$

$i=1$; $i=2$; $i=4$ (only three times)

$T(n) = \log n + \log n - 1$; $T(n) = 2\log n - 1$; $T(n) = O(\log n)$

∴ Whenever a (~~Step~~) step variable is multiplied by a constant number then it reaches n very faster so time complexity of such type of loop will be $O(\log n)$

```
for (i = n; i > 0 ; i/=c){
   Print("IBIT"):
}
```

$\log n$

$\log n - 1$

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

$T(n) = 2\log n - 1$; $T(n) = O(\log n)$

When the step variable is divided by a constant number then it reaches to zero very fast and it is not linear, so in such type of loop the time complexity will be $O(\log n)$

```
while (a >= b){
   a = a - b;
   count++;
}
```

$a=2, b=2$ (1 times) ; $a=4, b=2$ (2 times) ; $a=6, b=2$ (3 times)

$a=8$; $b=2$ (4 times)

2^1	2^2	2^3	2^4	2^k
1 time	2 time	3 time	4 time	n time

then $n = 2^k$ (Take log on b.s)

$k = \log n$; $T(n) = O(\log n)$

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

— n

— $n^2(n-1)$

$(n^2-1)(n-1)(n/2)$

$(n^2-1)(n-1)(n/2-1)$

$J=1$, $i=1$, $c=n/2 \times 1$	$J=2$, $i=4$, $c=n/2 \times 4$	$j=3$, $i=9$, $c=n/2 \times 9$	$J=4$, $i=16$, $c=n/2 \times 16$	$J=n$, $i=n^2$, $c=n/2 \times n^2$

$T(n) = n/2 \times 1 + n/2 \times 4 + n/2 \times 9 + n/2 \times 16 + n/2 \times n^2$

Taking $n/2$ common.

$T(n) = n/2\,(1+4+9+16 \ldots 2^k)$; $T(n) = n/2\,(1+2^2+3^2+4^2 \ldots 2^k)$; $T(n) = n/2\left(\frac{n(n+1)(2n+1)}{6}\right)$

$T(n) = O(n^4)$