Assignment-1

1)

Steps	j	Steps	i
0	1	0	n/2
1	2	1	n/2+1
2	3	2	n/2+2
⋮	⋮	⋮	n/2 + k < n
k	$k + \frac{n}{2} < n$	k	

$2k + n < 2n$ $\quad\quad$ $n + 2k < 2n$

$2k < n$ $\quad\quad$ $2k < n$

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

2)
```
void function(int n){
    if(n==1) return 0;        → O(1)
    for(int i=1; i<n; i++){
        for(int j=1; j<n; j++){
            printf("*");        } O(n)   } O(n)
            break;
        }
    }
```
O(n)

3)
```
void fun(int n){                T(O(√n))
    int i=1, count=0;
    for(i=1; i*i<=n; i++)
    {
        count++;
    }
}
```

4)

```
function(int n){
    for(int i=1; i<=n; i++)
    {
        for(j=1; j<=n; j*=2)
        {
            printf("*");
        }
    }
```

O(n)   O(log₂n)

$\text{Total} = O(n) \times O(\log_2 n) = O(n \log_2 n)$

5)

```
function(int n){
    for(int i=1; i<=n/3; i++){
        for(int j=1; k<=n; j+=4){
            printf("*");
        }
    }
```

O(n)   O(n/3)

$T(n) = O\left(\frac{n^2}{3}\right) = O(n^2).$

6)

```
function(int n){
    int sum=0, j=0;
    for(int i=0; i<n; i++){
        if(i>j) sum+=1;
        else{
            for(int k=0; k<n; k++){
                sum-=1;
            }
```

$O(n) + O(n) = O(n)$

* calculate order of complexity using master's theorem

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

$a = 3$ $3 < 2^2$

$b = 2$

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

5) $T(n) = 2T(n/2) + n\log n$

$a = 2$ 2^1 2^1

$b = 2$ $2 = 2$

$k = 1$

$p = 1$ $O(n\log^2 n)$

2) $T(n) = 4T(n/2) + n^2$

$a = 4$ a b^k

$b = 2$ 4 2^2

$k = 2$ $4 = 4$

$p = 0$ $O(n^2\log n)$

6) $T(n) = 2T(n/2) + n/\log n$

$a = 2$ $2 = 2^1$

$b = 2$

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

$p = -1$

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

$a = 1$ $1 < 4$

$b = 2$

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

$p = 0$

7) $T(n) = 2T(n/4) + n^{0.51}$

$a = 2$ $2 < 4^{0.51}$

$b = 4$

$k = 0.51$ $O(n^{0.51})$

$p = 0$

4) $T(n) = 16T(n/4) + n^1$

$a = 16$ $16 > 4$

$b = 4$

$k = 1$ $O(n^2)$

$p = 0$

8) $T(n) = 6T(n/3) + n^2\log n$

$a = 6$

$b = 3$ $O(n^2\log n)$

$k = 2$

$p = 1$

9) $T(n) = 7T(n/3) + n^2$

$a = 7$ $\quad 7 < 3^2$

$b = 3$ $\quad O(n^2)$

$k = 2$

$p = 0$

(5) $T(n) = 2T(n/2) + n$

$a = 2$

$b = 2$

$k = 1$

$a = b$

$O(n \log n)$

10) $T(n) = 4T(n/2) + \log n$

$a = 4$ $\quad 4 > 2$

$b = 2$ $\quad O(n^2)$

$k = 0$

$p = 1$

12) $T(n) = 3T(n/2) + n$

$a = 3$ $\quad 3 > 2$

$b = 2$ $\quad O(n^{\log_2 3})$

$k = 1$

13) $T(n) = 3T(n/4) + n \log n$

$a = 3$ $\quad 3 < 4$

$b = 4$ $\quad O(n \log n)$

$k = 1$

$p = 1$

14) $T(n) = 3T(n/3) + n/2$

$a = 3$ $\quad a = b$

$b = 3$ $\quad O(n \log n)$

$k = 0$

$p =$

Q-3 →

1) $f(n) = n \log n$

$g(n) = n^{\log n}$

$g(n) > f(n)$

2) $f(n) = 2^{\log n}$

$g(n) = n^{\sqrt{n}}$

$g(n) > f(n)$

3) $f(n) = 2^n$

$g(n) = 2^{2n}$

$g(n) > f(n)$

Date
Page

Q-4 1) $T(n) = \begin{cases} T(n-1) + n, & n>0 \\ 1, & n=0 \end{cases}$

$T(n) = T(n-1) + n$ — (i)

substuting $n = n-1$

$T(n-1) = T(n-2) + n - 1$ — (ii)

putting (ii) in (i)

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

$= T(n-2) + 2n - 1$ — (iv)

substituting $n = n-2$ in (i)

$T(n-2) = T(n-3) + n - 2$ — (iii)

putting (iii) in iv

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

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

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

⋮

k steps

$T(n) = T(n-k) + kn - k$

in Base case,

$T(n-k) = T(0)$

$n = k$

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

$T(n) = T(0) + n^2 - n$ ∴ $T(n) = n^2 - n = O(n^2)$

2) $T(n) = 2T(n-1) + 1$ —(1)

substituting $n = n-1$

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

putting $T(n-1)$ in 1

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

$= 4T(n-2) + 3$ —(ii)

putting $T(n-2)$ in (1)

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

putting $T(n-2)$ in (ii)

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

$= 8T(n-3) + 7$

!

k - steps

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

In Base case

$T(n-k) = T(0)$

$n = k$

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

$= 2^n \times 0 + 2^n - 1$

$= 2^n - 1$

$O(n) = 2^n$

Date
Page

3)
```
void test(int n){
    if (n>1){
        printf("ICT");
        test(n/2);
    }
}
```

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

According to Master's Theorem

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

$b = 2$

$k = 0$

4) $T(n) = \begin{cases} 2T(n/2) + n & ; n > 1 \\ 1 & ; n = 1 \end{cases}$

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

$a = 2$ $\quad 2^1 = 2^1$

$b = 2$ $\quad O(n \log n)$

$k = 1$