Q1 a

```
int s, i, n          3
cin >> n             1
s = 0                1
for ( i = n ; i >= 1 , i -- )
        s++
```

Counts on the for line: $i = n$ (1), $i >= 1$ ($n+1$), $i--$ (n); s++ (n)

$T(n) = 3n + 7$

$T(n) = O(n)$

$3n + 7 <= kn$

$3 + \frac{7}{n} <= k$

$n_0 = 1 \quad k >= 10$

$T(n) <= 10n$

Hence $T(n) = O(n)$

$n \geq 1$

Q1/b

```
int sum, i, j, n     4
  sum = 0            1
  cin >>             1
for ( i = 1 ; i < n ; i = i x 2
```

Counts on the for line: $i = 1$ (1), $i < n$ ($\log_2 n$), $i = i \times 2$ ($\log_2 n$)

$\log_2 n$ $\log_2 n$

for ($j = 1$; $j < n$, $j = j \times 2$)

Sum ++ ; $\log_2 n$

$T(n) = 7 + \log_2 n + \log_2 n + \log_2 n$

$+ 3\log_2 n \times \log_2 n$

$T(n) = 7 + 3\log_2 n + 3\log_2^2 n$

$T(n) < K \log_2^2 n$

$|T(n)| = 7\log_2^2 n + 3\log_2^2 n + 3\log_2^2 n$

$13 \log_2^2 n$

$n_0 > 1$ and $K >= 13$

$T(n) <= 13 \log_2^2 n$

$= O(\log_2^2 n)$

Q1/C

```
int Sum, i, j, k, n            5
  Sum = 0                      1
  k = 0                        1
  cin >> n;                    1
        1          n+1          n
  for(i=0 ,    i<n ,      ++i

      k = 0          n
      Sum++;         n
        n       n(n/3)+1   n(n/3)
      for(j= n;   j>0;    j=j-3)
          Sum++;   n(n/3)
          k=k*Sum; n(n/3)
          cout<< k; n(n/3)
      }
       cout << k;   n
```

$T(n) = 5n\left(\frac{n}{3}\right) + 6n + 9$

$T(n) <= kn^2$

$T(n) <= \left(\frac{5}{3} + 6 + 9\right)(n^2) = \left(\frac{50}{3} n^2\right)$

$n_0 >= 1$

$O(n^2)$ $\qquad$ $n_0 >= 1$

Q1 / d

```
int Sum, i, J, k, n          (5)
  sum = 0                    (1)
  cin >> n                   (1)
     (1)    (log2 n)          (log2 n)
for(i = 1,  i < n ;  i = i x 2
  (log2 n)   (n x log2 n)   (n x log2 n)
for(j = 0,  j < n ;  ++ j)
  (n x log2 n) (n log2^2 n)   (n log2^2 n)
for(k = 1,  k <= n,  k = k x 2
        sum ++ ;   (n log2^2 n)
```

$T(n) = 3n\log_2^2 n + 3\log_2 n + 3\log_2 n$

$+ 8$

$T(n) <= kn\log_2^2 n$

$(3 + 3 + 3 + 8)\, n \log_2^2 n$

$T(n) <= 17 n \log_2^2 n$

$O(n \log_2^2 n)$

$n_0 > 1$

Q1/e

```
int i, j, Sum, n                    4
cin >> n                            1
     1        log2n+1        log2n
for(i=1, i<=n, i=i×2
      cout << i ;        log2n
      Sum = 0 ;       log2n
        log2n
   for(j=1 ; j<=i, j=j×2
      Sum++ ;
      cout << i ;            log2n(log2n+1)/2
   }
   cout << sum          log2n
```

$T(n) = 4 \log_2 n(\log_2 n + 1) + 6 \log_2 n + 6$

$T(n) = 4 \log_2^2 n + 10 \log_2 n + 6$

$T(n) <= \log_2^2 n (4 + 10 + 6)$

$20 \log_2^2 n \qquad n > 1$

$k \geq 20$

$O(\log_2^2 n)$

Q2

Insertion sort $8n^2$
merge sort $64n \lg n$

$8n^2 \le 64n \lg n$

$n \le 8 \lg n$

$\frac{n}{8} \le \lg n$

$2^{\frac{n}{8}} = 2^{\lg n}$

$\left(2^{\frac{n}{8}}\right)^8 = (n)^8$

$2^n = n^8$

$2 = n^{\frac{8}{n}}$

$n \le 43$

Q3 $100n^2 = 2^n$

By trial and error and with graph help,

$n = 15$

Q4 $\Theta(n^3)$

$T(n) = \frac{1}{8}n^3 - 5n^2$

$K_2 n^3 \le T(n) \le K_1 n^3$

$\frac{1}{9}n^3 \le T(n) \le (\frac{1}{8}+5)n^3$

$K_2 = \frac{1}{9}$ $K_1 = \frac{13}{8}$

$n_0 \geq 1$

Q5 Selection Sort best case and worst case is $O(n^2)$ hence average case is $\Theta(n^2)$

```
Void InsertionSort(array[ ], size){
    for(J=1 TO n-1)
        smallest = j
    for(i = j+1 TO n)
        if array[i] < array[smallest]
            then smallest = i
        swap(Array[J], array[smallest])
    }
}
```

last element is already become sorted if $n-1$ are sorted.

Q6 $T(n) = 2m + 3m + 4m = P-1$

a) $m(2+3+4 \cdots n)$

$m\left(\frac{P-1}{2} + 4 + (P-2)\right)$

$m\left(\frac{P-1}{2}\right)(P+2) = O(mP^2)$

B) it is possible when two subarrays are merged at each level. e.g there are A, B, C, & D arrays with each having 5 elements then $P = A+B$ and $Q = C+D$ will be performed and $P+Q$ will be added.

It will reduce the time complexity to $O(mP)$

1)

if $y = 1$
return x (1)

otherwise
$c := x \times x$ (1)
ans := findpower [$y/2$]

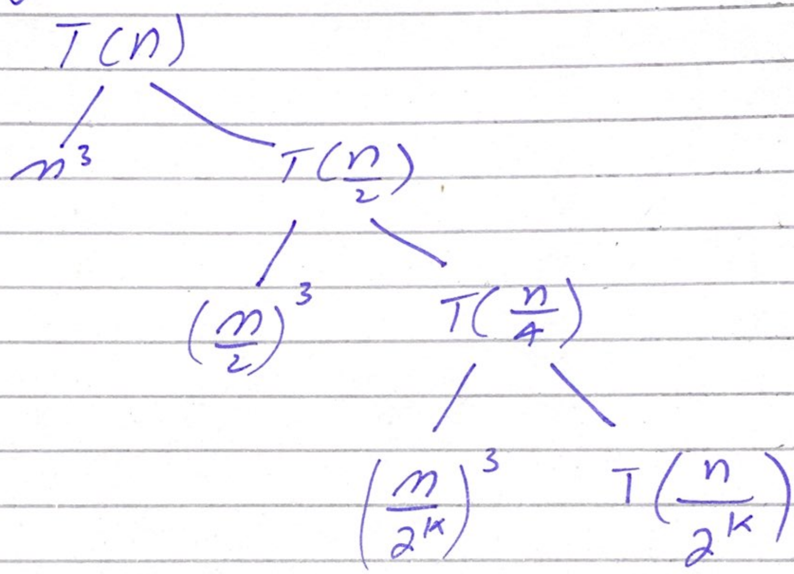
if y is odd
return $x \times$ ans (1)
otherwise return ans

$$T(n) = \begin{cases} \cancel{\text{@}} = O(1) & y = 1 \\ T\left(\frac{n}{2}\right) + 2 & y > 1 \end{cases}$$

Q8/a

$$T(n) = T(\tfrac{n}{2}) + O(n^3)$$

Using Tree method



$$T(n) = n^3 + \frac{n^3}{8} + \frac{n^3}{64} \cdots \frac{n^3}{8^k}$$

$$T(1) = \frac{n^3}{8^k} \qquad n^3 = 8^k$$

$$k = \log_8 n$$

$$n^3 = 2^{3k} \Rightarrow n = 2^k$$

$$k = \log_2 n = \lg n$$

$$T(n) = n^3\left(1 + \frac{1}{8} + \frac{1}{64} \cdots \frac{1}{8^k}\right)$$

Geometric series with $r = \frac{1}{8}$

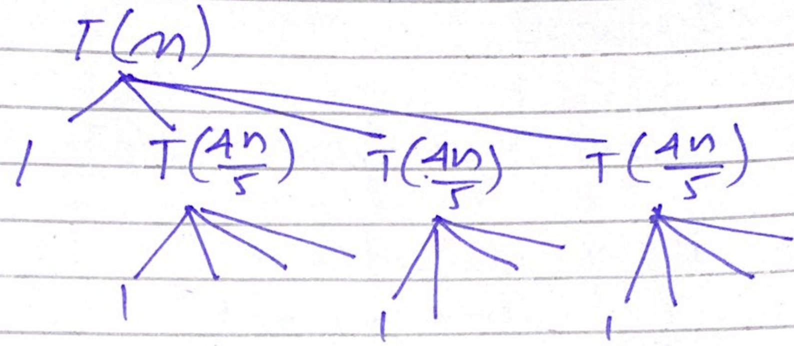
$$\sum_{k=0}^{\infty} \frac{1}{8^k} = \frac{a}{1-r} = \frac{1}{1-\frac{1}{8}} = \frac{8}{7}$$

$$T(n) = \frac{8}{7} n^3$$

$O(n^3)$ Ans.

1 8/b

$$T(n) = 3T\left(\frac{4n}{5}\right) + O(1)$$



$$S = 3^0 + 3^1 + 3^2 \dots 3^k$$

$$T\left(\frac{4^k n}{5^k}\right) = T(1)$$

$$4^k n = 5^k$$

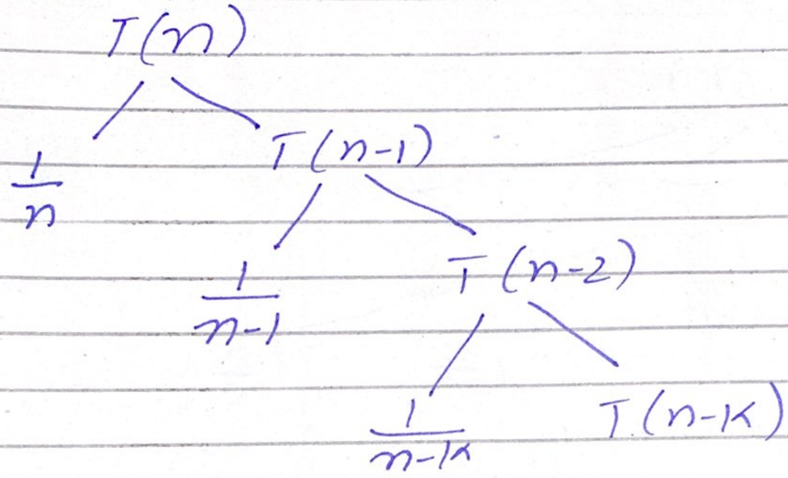
$$n = \left(\frac{5}{4}\right)^k \Rightarrow k = \log_{\frac{5}{4}} n$$

$$S = \frac{1\left(3^{\log_{\frac{5}{4}} n} - 1\right)}{3-1}$$

$$= \frac{n^{\log_{\frac{5}{4}} 3} - 1}{2}$$

$$O(\log(n^4))$$

Q 8/c $T(n) = T(n-1) + \frac{1}{n}$



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

$n-k = 1$

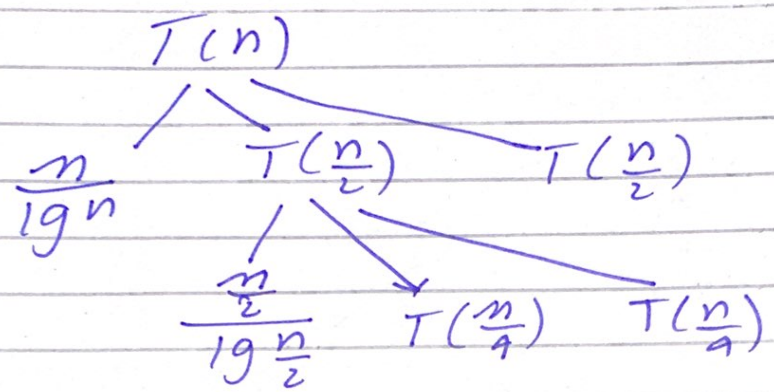
$T(n) = \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \dots \frac{1}{n}$

In reverse order.

$T(n) = \ln(n) + O(1)$

$= O(\ln(n))$

Q8/d $T(n) = 2T\left(\frac{n}{2}\right) + \frac{n}{\lg n}$



$= 2^k \left(\frac{n}{2^k}\right) / \lg\left(\frac{n}{2^k}\right)$

$\frac{n}{\lg\left(\frac{n}{2^k}\right)} = 1$

$\cancel{n}\, 1 = \lg\left(\frac{n}{2^k}\right) = \lg n - \lg 2^k = 1$

$\lg n = \lg 2^k$

$k = \lg n$

$T(n) = \frac{n}{\lg n} + \frac{n}{\lg n - \lg 2} + \frac{n}{\lg n - \lg 4} \dots$

$\frac{n}{\lg n - \lg 2^k}$

$$T(n) = n\left(\frac{1}{\lg n} + \frac{1}{\lg n - \lg 2} \cdots \frac{1}{\lg n - 2^k}\right)$$

$$= n\left(\frac{1}{\lg n} + \cdots \frac{1}{2} + \frac{1}{1}\right)$$

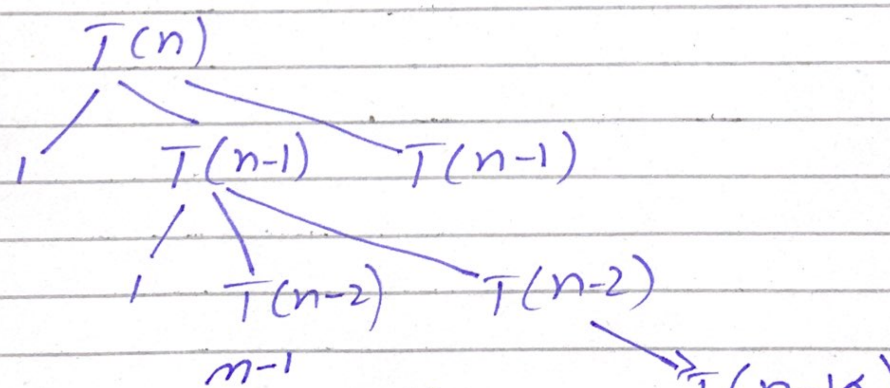
$$= n\left(\frac{1}{1} + \frac{1}{2} + \frac{1}{3} \cdots \frac{1}{\lg n}\right)$$

$$= n \ln(\lg n)$$

$$O(n \ln(\lg(n))$$

or $O(n \lg^2 n)$

Q 8/e $\quad T(n) = 2T(n-1) + \Theta(1)$



$$T(n) = \sum_{k=0}^{n-1} 2^k$$

$T(n) = 1 + 2 + 4 \cdots 2^k$

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

$n - k = 1$

$k = n - 1$

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

Geometric Serses

$r = 2 \qquad n \Rightarrow n-1$

$S = \frac{1(2^{n-1} - 1)}{2-1}$

$= \frac{2^n - 1}{2} = O(2^n)$