Kongu Engineering College

School of Communication and Computer Sciences

Department of Computer Scinece and Engineering

FOLLOWUP ACTIONS FOR THE FAILED STUDENTS

Module Test - I

Date of Exam : 20.01.2018

Class : II & IV & B sec

Max Marks : 10

Subject Code : 14

Name of the Subject : Design and Analysis of Algorithms

Sl. No	Register No	Student Name	Remedial measures (Assignment, special session, etc...)	Sl. No	Register No	Student Name	Remedial measures (Assignment, special session, etc...)
1	16CSL238	KAYALVIZHI C	Assignment				
2	16CSL239	MOBIN P BENNY					
3	16CSL240	PAVITHRA A					
4	16CSR060	GAYATHRI A					
5	16CSR062	GOKUL S					
6	16CSR067	HARIHARAN V P					
7	16CSR068	HARRISH S J					
8	16CSR077	JAIVIGNESH D					
9	16CSR081	JEEVANANTH S					
10	16CSR087	KARTHICK M					
11	16CSR089	KARTHIK M					
12	16CSR094	KAVIN M					
13	16CSR096	KAVINKUMAR M					
14	16CSR100	KAVYA S					
15	16CSR101	KAYALVIZHI R					
16	16CSR108	MAHENDRAN A P					

K. Kowsalya

Name of the Subject I/C

Signature of Subject I/C

Rajal 6/2

Signature of

R KARTHIKEYAN

16CSR091

DAA ASSIGNMENT

11) Quick sort Average case

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

$T(q) = \frac{1}{n}T(1) + \frac{1}{n}T(2) + \frac{1}{n}T(3) \cdots \frac{1}{n}T(n-1)$

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

$T(n) = \frac{1}{n}\sum_{j=0}^{n-1} T(j) + \frac{1}{n}\sum_{j=0}^{n-1} T(j) + n$

$T(n) = \frac{2}{n}\sum_{j=0}^{n-1} T(j) + n \longrightarrow (1)$

multiply by n on both side

$nT(n) = 2\sum_{j=0}^{n-1} T(j) + n^2 \longrightarrow (2)$

sub $n = n-1$ in eq (2)

$nT(n) = 2\sum_{j=0}^{n-1} T(j) + (n-1)^2 \longrightarrow (3)$

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

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

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

$nT(n) = (n+1)T(n-1) + 2n \longrightarrow (4)$

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

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

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

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

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

In general

$$\frac{T(n)}{n+1} = \frac{T(1)}{2} + \frac{2}{3} + \frac{2}{4} + \frac{2}{5} \cdots \frac{2}{n-1} + \frac{2}{n}$$

$$\frac{T(n)}{n+1} = \frac{T(1)}{2} + 2 \sum_{j=3}^{n+1} \frac{1}{j-1}$$

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

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

$$= 0 + \sum_{j=1}^{n+1} \frac{1}{j}$$

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

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

$$T(n) \simeq (n+1)\log(n+1)$$

$$T(n) \in O(n \log n)$$

A11 A12 B11 B12

$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ $\begin{pmatrix} 4 & 3 \\ 2 & 1 \end{pmatrix}$

A21 A22 B21 B22

$S1 = 3-1 = 2$

$S2 = 1+2 = 3$

$S3 = 3+4 = 7$

$S4 = 2-4 = -2$

$S5 = 1+4 = 5$

$S6 = 4+1 = 5$

$S7 = 3-4 = -1$

$S8 = 2+1 = 3$

$S9 = 2-3 = -1$

$S10 = 4+3 = 7$

$P1 = 1.(2) = 2$

$P2 = 3(1) = 3$

$P3 = 7(4) = 28$

$P4 = 4(-2) = -8$

$P5 = 25$

$P6 = -3$

$P7 = -14$

$= \begin{bmatrix} -3-8+25-3 & 2+3 \\ 28-8 & 2-28+25+14 \end{bmatrix}$

$= \begin{bmatrix} 11 & 5 \\ 20 & 65 \end{bmatrix}$

12) i)

```
Insertion sort (A[0...n-1]
    input  A[0...n-1] of n
    output A[0...n-1]
        for i ← 1 to n-1 do
            v ← A[i]
            j ← i - 1
            while j ≥ 0 and A[j] > v do
                A[j+1] ← A[j]
                j ← j - 1
            A[j+1] ← v
```

13) Brute force (V[a,e,i,o,u]
All vowels (V[a,e,i,o,u])
{ count = 0
for [i=1 to n]
{ If (a[i] = 'a' | 'e' | 'i' | 'o' | 'u'
count = count + 1
}
return count
}

$$T(n) = \sum_{i=1}^{n} 5$$

$$= 5 \sum_{i=1}^{n} 1 \qquad \geq n$$

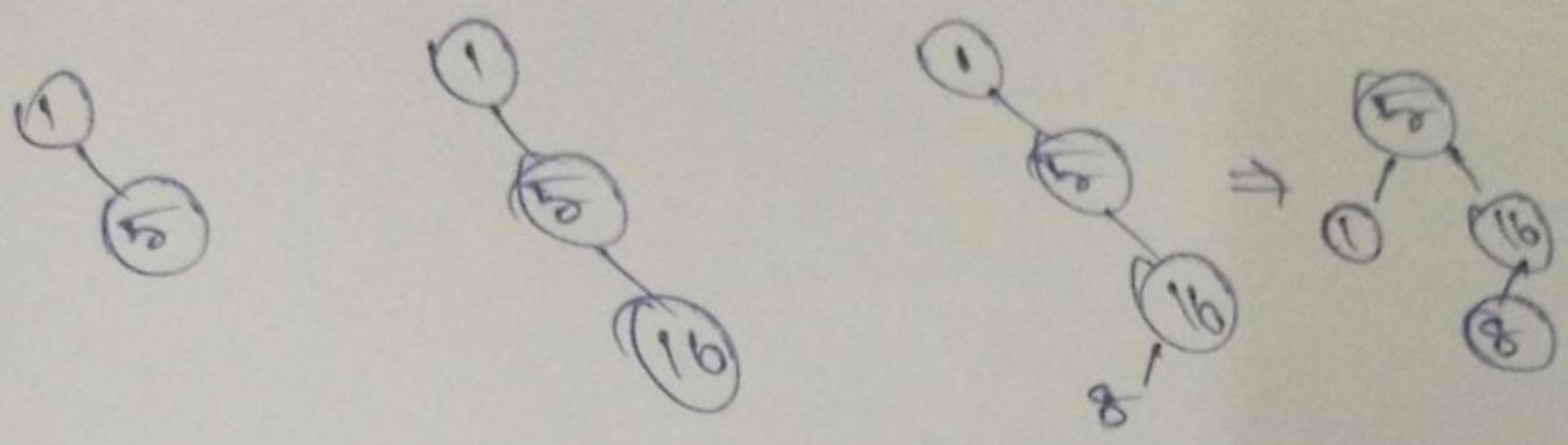
$$= 5(n-1+1) \qquad T(n) \in O(n)$$

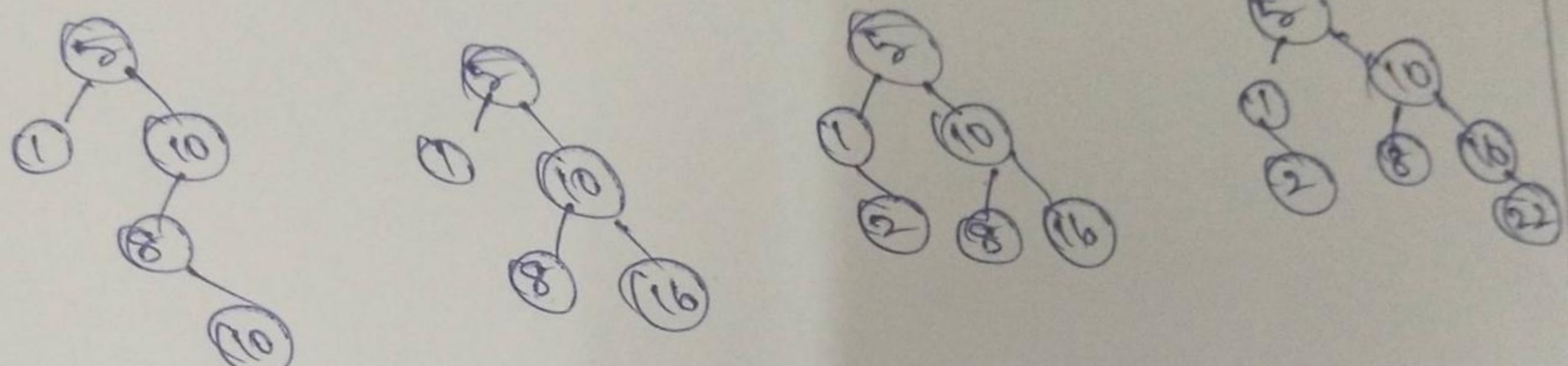
$$= 5n$$

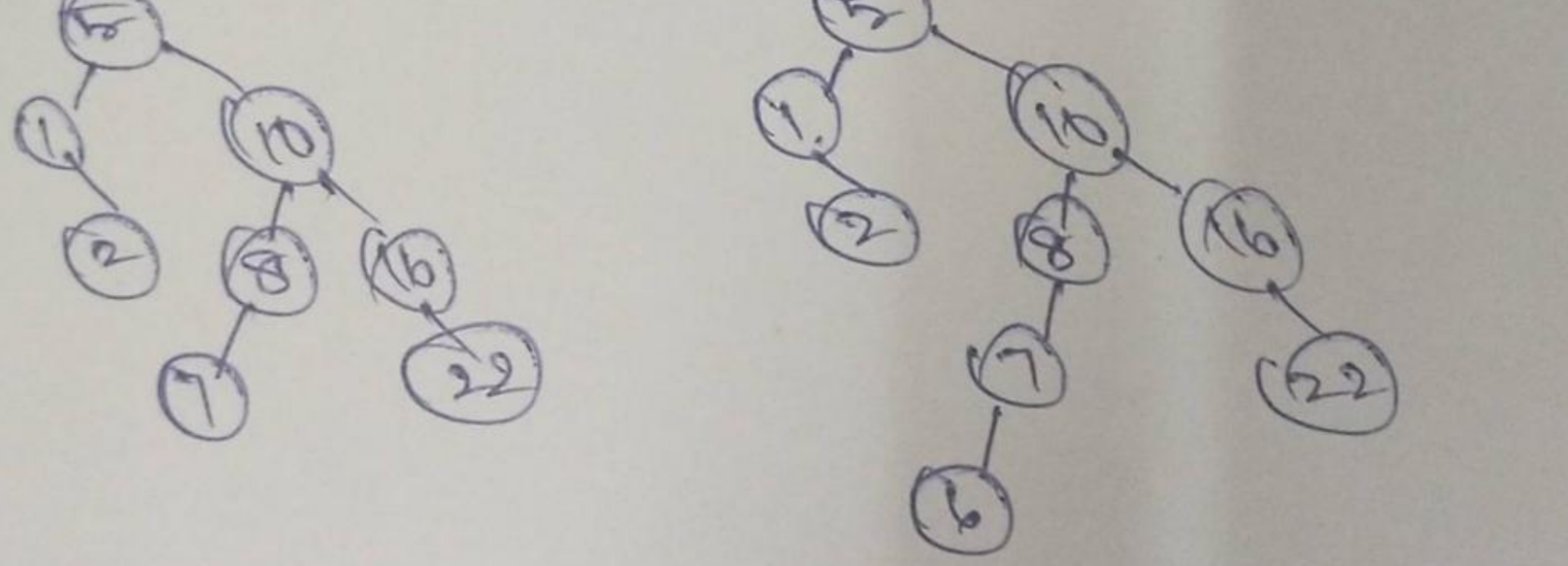
```
Presorting (A[0...n-1])
  Input A[0...n-1]
{
  return true
  else
     False
  }
  for i<-0 ton-2
{ do
  if A[i]= A[i+1]
    return false
  return true


Alg mergeSort (A[0...n-1])
{
  If
    n>0
{
  copy A[0..n/2] to B[0...n/2]
  copy A[n/2+1...n-1] to C[0...n/2]
  mergeSort (B[0...n/2])
  mergeSort (C[0...n/2])
  merge (B,C,A)
}
}
```

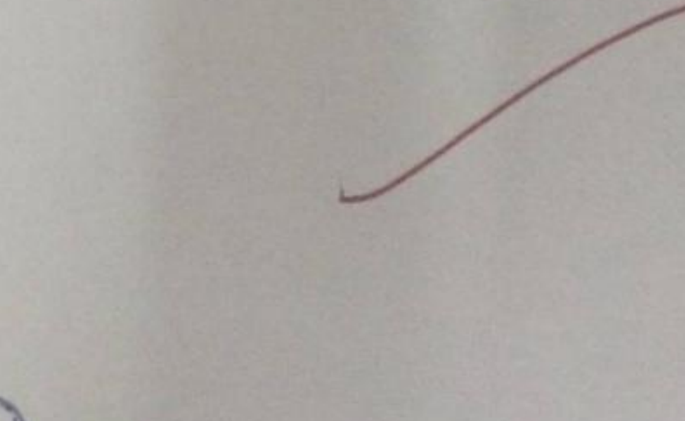
AVL tree

1, 5, 16, 8, 10, 2, 22, 7, 6, 1









Kongu engineering College

School of Communication and Computer Sciences

Department of Computer Scinece and Engineering

FOLLOWUP ACTIONS FOR THE FAILED STUDENTS

Module Test - II

Date of Exam : 28.02.17 Max Marks : 100

Class : II & IV & B sec Subject Code : 14CS/IS

Name of the Subject : **Design and Analysis of Algorithms**

Sl. No	Register No	Student Name	Remedial measures (Assignment, special session, etc...)
1	16CSL238	KAYALVIZHI C	Assignment
2	16CSL239	MOBIN P BENNY	
3	16CSL240	PAVITHRA A	
4	16CSR060	GAYATHRI A	
5	16CSR063	GOKULAPRIYA M	
6	16CSR068	HARRISH S J	
7	16CSR074	HINDHUJA K M	
8	16CSR077	JAIVIGNESH D	
9	16CSR081	JEEVANANTH S	
10	16CSR089	KARTHIK M	
11	16CSR091	KARTHIKEYAN R	
12	16CSR096	KAVINKUMAR M	
13	16CSR100	KAVYA S	

Sl. No	Register No	Student Name	Remedial measures (Assignment, special session, etc...)

Name of the Subject I/C

Signature of Subject I/C

Year in charge 9/3/18

Signature of HoD

Transform and conques

types
- → instance simplification → simplify and solving the problem
- → representation change
- → problem reduction

General plan

convert the problem into another type and solve

ex :- heap sort

mode

Repeated nos more times called as mode

Algorithm :-

```
Alg mode (A[. . .])
{
count = 0 , num = A[0], count1 = 1

for (i=0 to n-1)
{
  if (A[i] = = A[i+1])

       count1++;

  else if (count1 > count)
  {
    count = count1
    Num = A[i];
    count1 = 1;
  else
  } count1 = 1;
}

  display (count, Num);
}
```

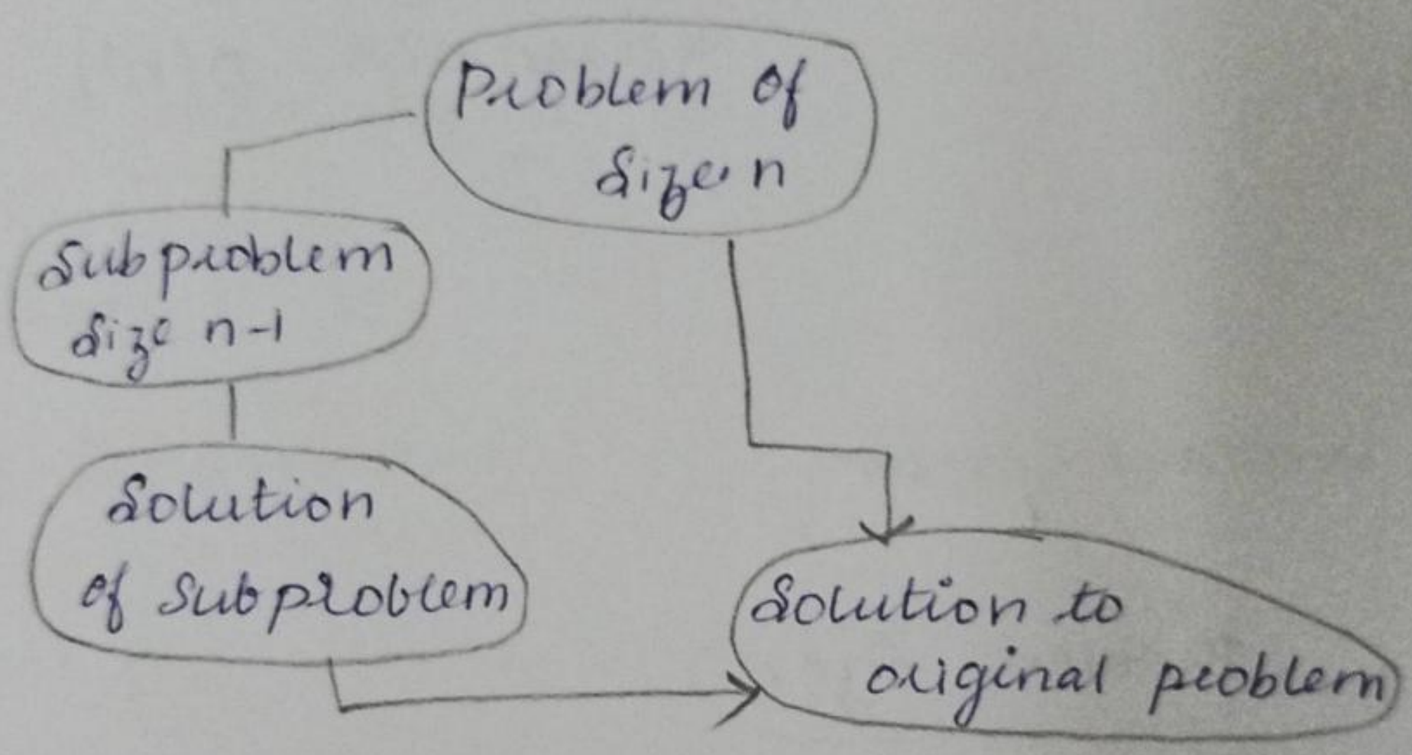
```
Alg uniqueness (n[], n)
{
  for (i:0 to n-2)
  {
    if (a[i] == a[i+1])
      return false;
  }
    return true;
}
```

3.

Types :-

* decrease by constant
  - sequential search, insertion sort, topological sort
* decrease by constant factor
  - binary search
* variable size decrease
  - Euclid algorithm

Decrease by constant :-



Eg :- Insertion Sort

	89	45	68	90	29	34	17
45	45	89	68	90	29	34	17
68	45	68	89	90	29	34	17
90	45	68	89	90	29	34	17
29	29	45	68	89	90	34	17
34	29	34	45	68	89	90	17
17	17	29	34	45	68	89	90

Algorithm:-

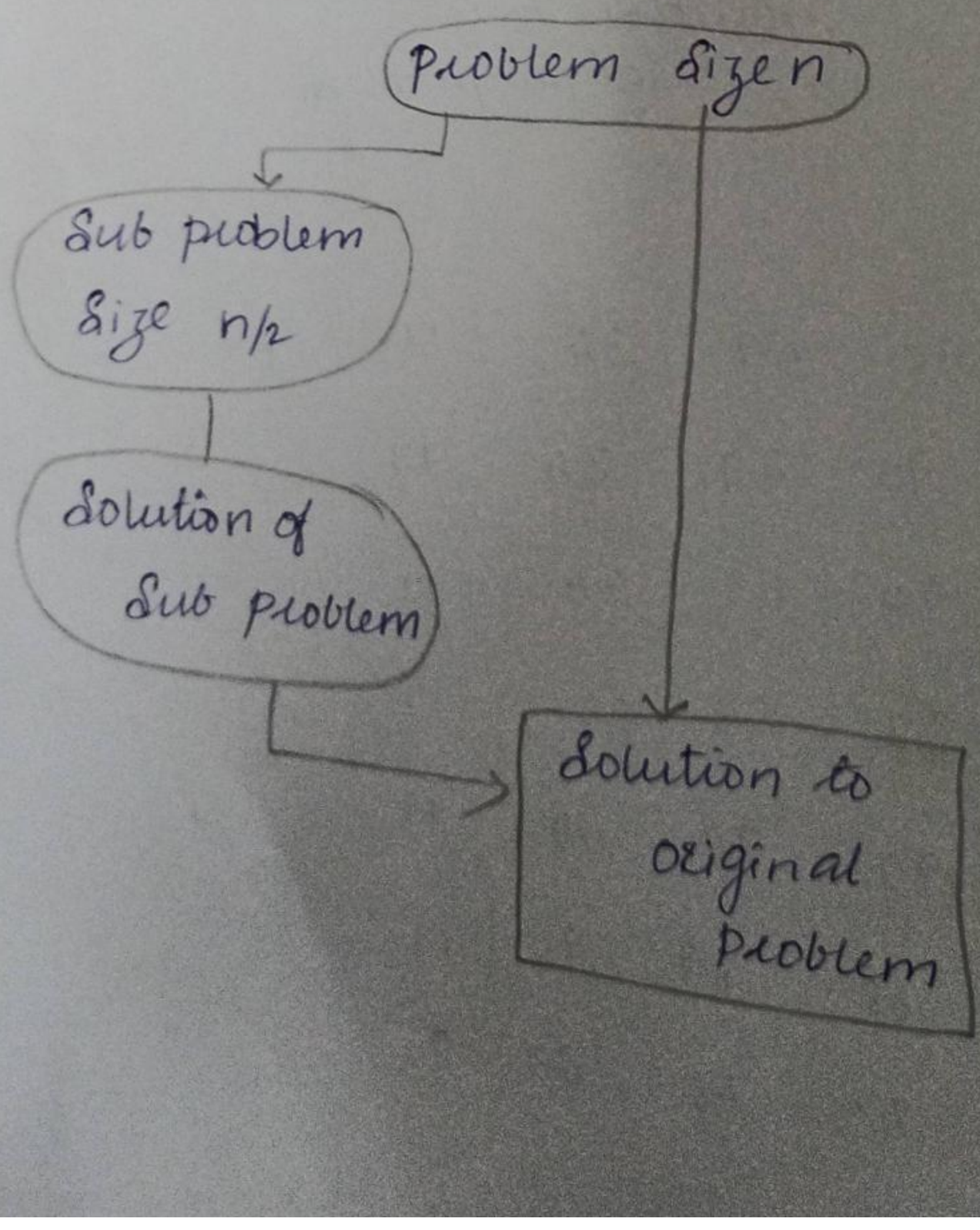
```
Alg insertionsort (A[0...n-1])
{
  for (i=1 to n-1)
  {
    v=A[i], j=i-1;
    while (j>=0 and A[j]>v)
    {
      A[j+1] = A[j];
      j=j-1;
    }
    A[j+1]=v
  }
}
```

Best case - $O(n)$

Worst case - $O(n^2)$

Average case - $O(n^2)$

Decrease by constant factor :-



Ex :- Binary Search - O(log(n))

0	1	2	3	4	5	6	7	8	9
10	25	38	59	100	120	158	200	800	850

$l = 0$

$r = 9$

$m = \frac{l+r}{2} = 9/2 = 4.5 = 5$

Since $A[5] < key \Rightarrow 120 < 800$

$l = m+1 \quad r = 9$

$m = \frac{6+9}{2} = \frac{15}{2} = 7.5 = 8$

$A[m] == key$

Algorithm :-

```
Alg binarysearch (A[0...n-1], key, l, r)
{
  m = (l+r)/2;
  if (l ≤ r)
  {
    if [A[m] == key)
        return m;
    else if (key > A[m])
        binary search (a, key, m+1, r);
    else if (key < A[m])
        binarysearch (a, key, l, m-1);
  }
  return -1;
}
```

Variable Size decrease :-

In variable size decrease each and every iteration is decrease by variable size.

eg:- Euclid algorithm

eg :- gcd (60, 24)

gcd(m, n) = gcd (n, m and n)

gcd [60, 24) = gcd (24, 12)

= gcd (12, 0)

Algorithm :-

```
int gcd ( int m, int n)
{
  if (n==0)
    return m;
  else
    t = m%n;
    gcd(n, t);
}
```