Audit Course-2

RCA-A02 Fundamental of Data Structure, Numerical and Computational Theory

UNIT-I

Arrays:- Array Definition, Representation and Analysis, Single and Multidimensional Arrays, Searching: Sequential search, binary search, comparison and analysis, Sorting: Insertion Sort, Bubble sort, Quick Sort, Two Way Merge Sort, Heap Sort.

Linked list:- Representation and Implementation of Singly Linked Lists, Two –way Header List, Traversing and Searching of Linked List, Overflow and Underflow, Insertion and deletion to/from Linked Lists algorithm (Beginning, end and middle).

UNIT-II

Binary Search Trees: Binary Search Tree (BST), Insertion and Deletion in BST, Complexity of Search Algorithm.

Curve fitting and Approximation: Method of least squares, fitting of straight lines, polynomials, exponential curves.

Regression analysis: Linear and Non-linear regression, multiple regressions

UNIT-III

Time series Analysis and Hypothesis Testing: forecasting models and methods. Test of significance, Chi-square test, t-test, F-Test

Finite State Machines (FSM): Introduction, Deterministic (DFA), Nondeterministic (NFA). Conversions and Equivalence: Equivalence between NFA with and without ε transitions. NFA to DFA conversion. Minimization of FSM.

UNIT-IV

Regular Expression & Regular Set: Definition, Properties, Pumping Lemma, and Decision problem for regular language.

Grammar: Introduction, Definition, Different types, Derivation Tree, Different Normal Forms, Ambiguous Grammar and its implications, Chomsky hierarchy. Different Classes of Languages.

Pushdown Automata (PDA): Definition, PDA and CFL (Context-Free Language), Acceptance of Strings.

Turing Machine: Introduction, Turing Machine Model.

References:-

1. S. Lipschutz, "Data Structures", Mc-Graw Hill International Editions.
2. K.L.P. Mishra, N. Chandrasekaran, "Theory of Computer Science", PHI.
3. Rajendra Kumar, "Theory of Automata, Languages and Computation", Mc-Graw Hill.
4. M. Goyal, "Computer-Based Numerical & Statistical Techniques", Infinity Science Press.

Creating the Array

Algorithm - Algorithm is a step by step solution of a problem

Algorithm for Creating the array :-

create_array(array[ ], size)

Step 1: I = 1

Step 2: While I <= size
Input array[I]
I = I+1
Endwhile

Step 3: End

(1)	5
(2)	20
(3)	15
(4)	10
(5)	60

Algorithm for traversing

traversing_array(array[ ], size)

Step 1: I = 1

Step 2: While I <= size
print array[I]
I = I+1
Endwhile

Step 3: End.

Program to create the array and traversing

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int a[100], i, size;
    clrscr();
    printf("Enter the size of Array max. 100");
    scanf("%d", &size);

    for(i=0; i<size; i++)
    {
        printf("Enter the Element ");
        scanf("%d", &a[i]);
    }

    for(i=0; i<size; i++)
    {
        printf("%d\n", a[i]);
    }
    getch();
}
```

output

```
Enter the size of Array max. 100  5
Enter the element  15
Enter the element  3
Enter the element  6
Enter the element  11
Enter the element  21
```

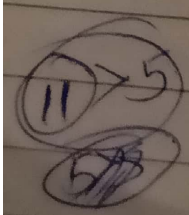
15
3
6
11
21

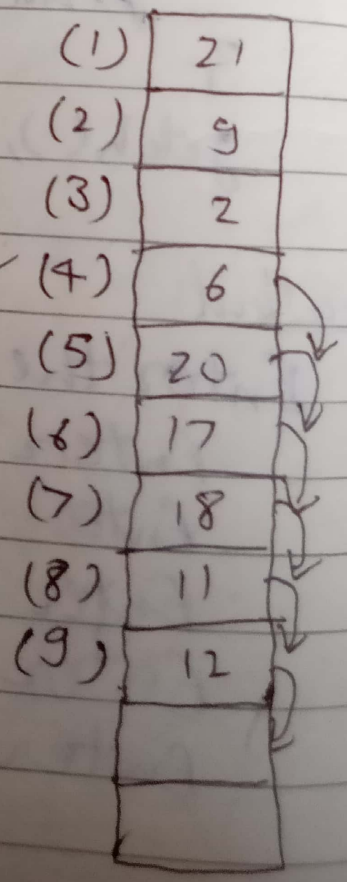
Inserting the element into an Array

Algorithm

Insert_array(array[], size, element, pos)

Step 1: I = size+1
Step 2: while I > pos
        array[I] = array[I-1]
        I = I-1
        Endwhile
Step 3: array[pos] = element
Step 4: End





Program :-

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int a[100], i, size;
    clrscr();
    printf("Enter the size of Array max. 100");
    scanf("%d", &size);

    for(i=0; i<size; i++)
    {
      printf("Enter the element");
      scanf("%d", &a[i]);
    }

    printf("Enter the position and element");
    scanf("%d %d", &pos, &element);

    i=size;
    while(i>pos)
    {
        a[i]=a[i-1];
        i--;
    }
    a[pos] = element;

    for(i=0; i<=size; i++)
    {
        printf("%d\n", a[i]);
getch(); }
```

Algorithm to insert an element into sorted list.

Step 1: I = 1
While I <= size
If element < array [I]
exit loop
endif
I = I + 1
Endwhile

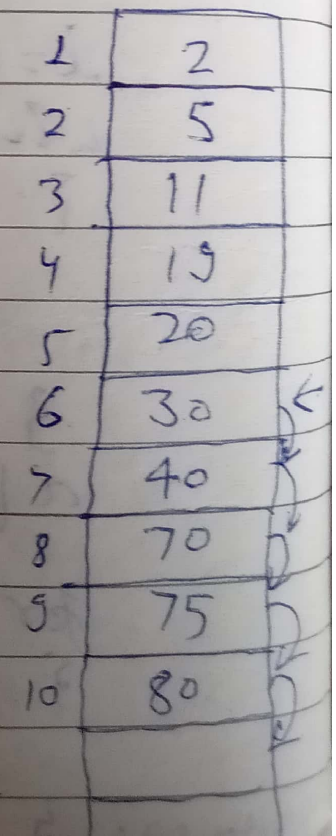
Step 2: pos = I

Step 3: I = size + 1

Step 4: While I > pos
array[I] = array[I-1]
I = I - 1
End while

Step 5: array[pos] = element

Step 6: End.



Deletion :-

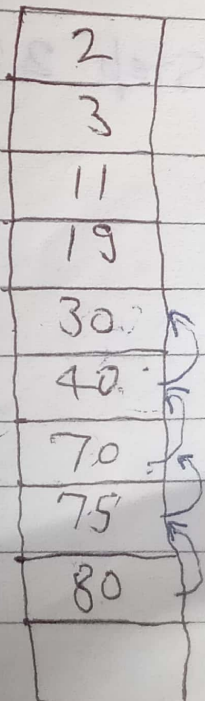
delete_element(array[], element, size)

Step 1: [search the element]
I = 1, found = 0
while I <= size
If element = array[I]
found = 1
exit loop
endif
I = I + 1
Endwhile

Step 2: If found = 0
print "Not found"
exit
endif

Step 3: while I <= size
array[I] = array[I+1]
I = I + 1
End while

Step 4: End.



Searching:- Searching means to search a particular element in the list. There are two main techniques for searching.

(i) Sequential Search OR Linear Search

(ii) Binary Search

(i) Sequential Search:- In this technique searched element will be compared from top to bottom. So, there is no need to sort the list.

Algorithm:-

Search-Linear(array[ ], element, size)

Step 1: I = 1, found = 0

Step 2: While I <= size

If array[I] = element
found = 1
exit loop

Endif
I = I+1
EndWhile

Step 3: If found = 1
print "Element is found"
else
print "Not found"
endif

Step 4: END.

Program:-

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int a[100], size, i, found=0;
    clrscr();
    printf("Enter the size");
    scanf("%d", &size);

    for(i=0; i<size; i++)
    {
        printf("Enter the element");
        scanf("%d", &a[i]);
    }
    printf("Enter the element to be search");
    scanf("%d", &element);
    for(i=0; i<size; i++)
    {
        if(a[i] == element)
        {
            found = 1;
```

```
            break;
        }
    }
    if(found==1)
        printf("Element is found");
    else
        printf(" Element Not found");
    getch();
}
```

```
Enter the size : 10
Enter the element : 21
                  : 10
                  : 5
                  : 4
                  : 22
                  : 45
                  : 18
Element is found.
```

Binary Search:- To use this technique the list must be in sorted order. In this technique the element will be compared with middle element of the list. If not found then check the element is greater than this middle element then we have to compare with next middle element of second half list.

Algorithm:-

binary_search(array[], size, element)

[Suppose array is in sorted order]

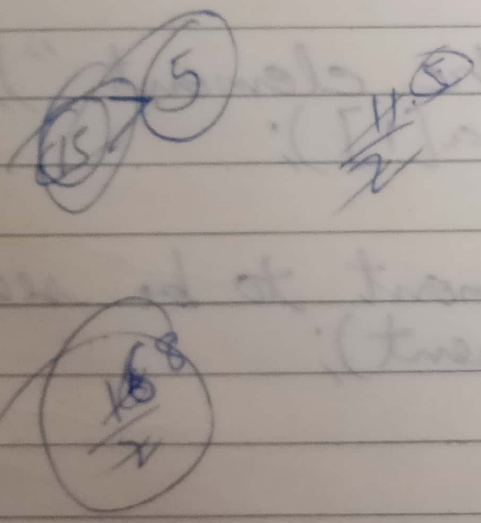
Step 1: beg = 1, end = size, found = 0

Step 2: while beg <= end
mid = (beg + end)/2;
if (array[mid] = element)
found = 1;
exit loop
endif
if (element > array[mid])
beg = mid + 1
else
end = mid - 1
endif

Endwhile.

Step 3: If found = 1
printf " Element found"
else
print " Not found "
endif

Step 4: End



(1)	2	beg = 1
(2)	5	
(3)	10	
(4)	15	end = mid-1
(5)	25	
(6)	30	beg = mid+1
(7)	40	
(8)	60	
(9)	75	
(10)	85	end = 10

Program of Binary Search :-

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[100], size, i, beg, end, mid, element;
    clrscr();
    /* Creating the Array */
    printf("Enter the size of Array");
    scanf("%d", &size);
    for(i=0; i<size; i++)
    {
        printf("Enter the element");
        scanf("%d", &a[i]);
    }
    printf("Enter the element to be search
    scanf("%d", &element);

    beg = 0;
    end = size-1;
    while(beg <= end)
    {
        mid = (beg+end)/2;
        if(a[mid] == element)
        {
            found = 1;
            break;
        }
        if(element > a[mid])
```

```
            beg = mid+1;
        else
            end = mid-1;
    }
    if(found == 1)
        printf("Element is found");
    else
nd = 0;     printf("Not found");

    getch();
}
```

2	a[0]
5	a[1]
11	a[2]
12	a[3]
30	a[4]
41	a[5]
55	a[6]
70	a[7]
81	a[8]
97	a[9]

Linked List

Linked List is a collection of nodes. Each node contains at least two parts. The first part is Information part which contains the data and another part is address part which contains the address of next node.

Characteristics of Linked List:-

(1.) It will be created in memory dynamically. So, there is no wastage of memory.

(2.) It is less expensive for Insertion & Deletion.

(3.) Processing is fast.

(4.) Address of each node will be randomly.

Types of Linked List:-

(1) Single Linked List (One-way List)

(2) Doubly Linked List (Two-way List)

(3) Circular Linked List

(1) Single Linked List :-

In Single Linked list, each node contains only one address part. Information part may be more than one.

Operations on Linked List :-

(1.) Create

(2.) Insertion

(3.) Deletion

(4.) Traversing

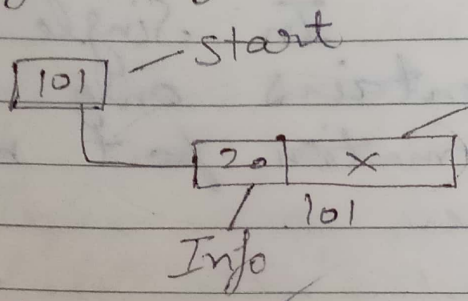
(5.) Searching

(6.) Sorting

(7.) Concatenation

Algorithm for creating a linked list having only one node:-

Ex. →



Create (start, element, avail)

Step 1: [check the available space]

If avail = NULL

print "Insufficient Memory"
exit
endif

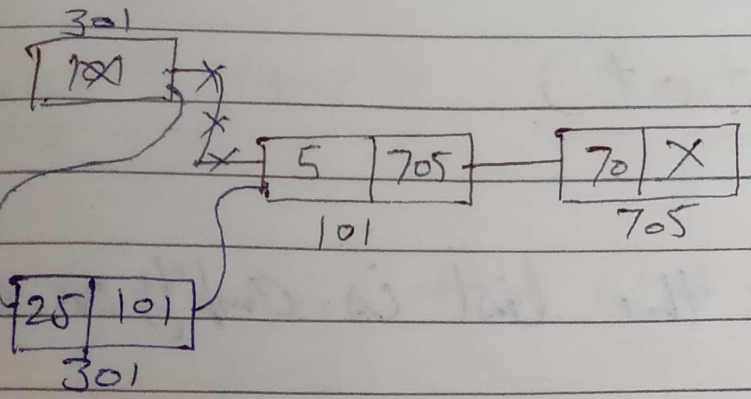
Step 2: [create a new node]

current = create a node
info[current] = element
link[current] = NULL

Step 3: start = current

Step 4: END.

Algorithm for Inserting a node in begining of Linked List:-



Insert_begining (start, element, avail)

Step 1: [check the available Space]

If avail = NULL

print "Insufficient Memory"
exit
endif

Step 2: [create a node]

current = Create a node
info[current] = element

Step 3: link[current] = start

Step 4: Start = current

Step 5: END

Algorithm for Traversing the Linked List.

traversing (start)

Step 1: [check the list is empty or Not]

if start = NULL

print "List is empty"
exit
Endif.

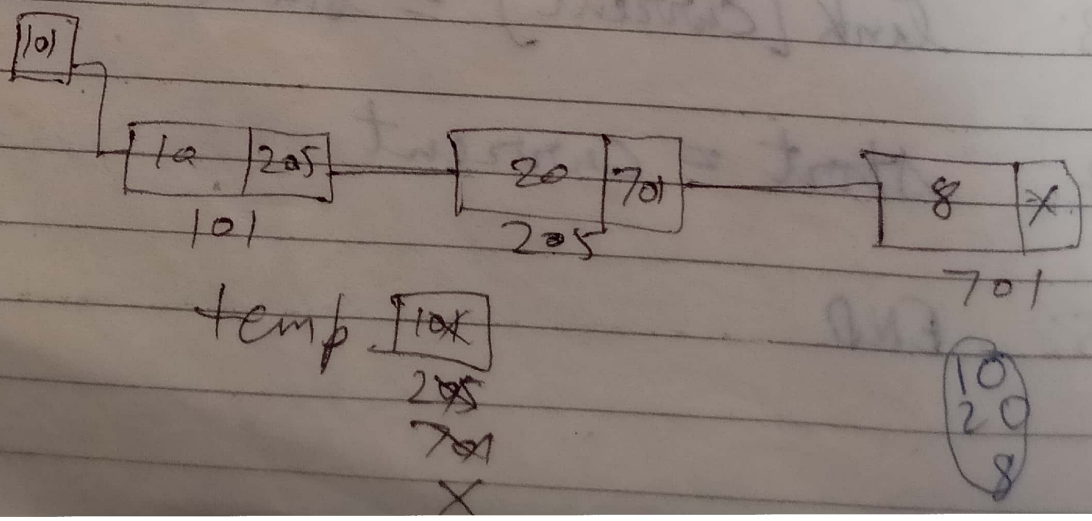
Step 2: temp = start

Step 3: while temp ≠ NULL

Print info[temp]
temp = link[temp]

End while

Step 4: End.



Algorithm to count total nodes in a Linked List:-

Count (start)

Step 1: [check the list is empty or Not]

```
if start = NULL
    print "List is empty"
    Exit
endif.
```

Step 2: temp = start, count = 0

Step 3: While temp ≠ NULL

```
    count = count + 1
    temp = link[temp]
EndWhile
```

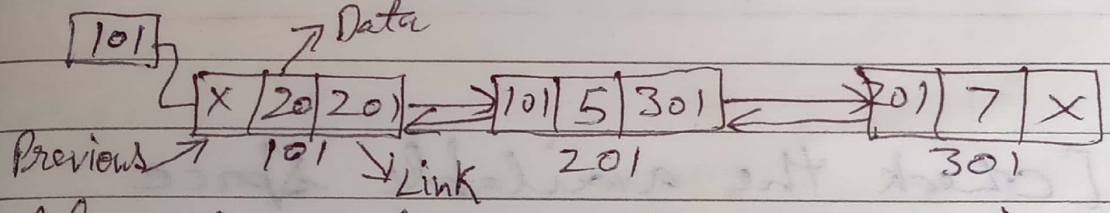
Step 4: Print count

Step 5: End

Doubly Linked List (Two-way List):-

In doubly linked list each node contains two address parts. The first part is previous part which contains the address of previous node. Another part is link part which contains the address of next node.

The information part may be one or more. The benefit of doubly linked list is that we can traverse in both direction.



Algorithm to Insert a Node in Beginning:-

insert_begining(start, element, avail)

Step 1: [Check the available space or Not]

```
If avail = NULL
    print "Insufficient Memory"
    exit
endif.
```

Step 2: [create a node]

```
current = create a node
info[current] = element
```

prev[current] = NULL
link[current] = start
prev[start] = current
start = current

Step 3: END.

Algorithm to Insert a Node at End.

insert_end (start, element, avail)

Step 1: [Check the available space or Not]

If avail = NULL
printf "Insufficient memory
exit
Endif.

Step 2: [Create a Node]
current = create a node
info[current] = element
link[current] = NULL

Step 3: [Find out last node's address]
temp = start
while link[temp] ≠ NULL

temp = link[temp]
Endwhile.

Step 4: link[temp] = current

Step 5: prev[current] = temp

Step 6: END.

Imp:

Algorithm to Insert a Node at desired position :-

insert_at_desired (start, avail, element, pos)

Step 1: [check the available space]
If avail = NULL
print "Insufficient Memory"
exit
Endif.

Step 2: [create a Node]

current = create a node
info[current] = element

Step 3: [check the position is first or Not]

If pos = 1

link[current] = start
prev[start] = current
prev[current] = NULL
start = current
exit
Endif.

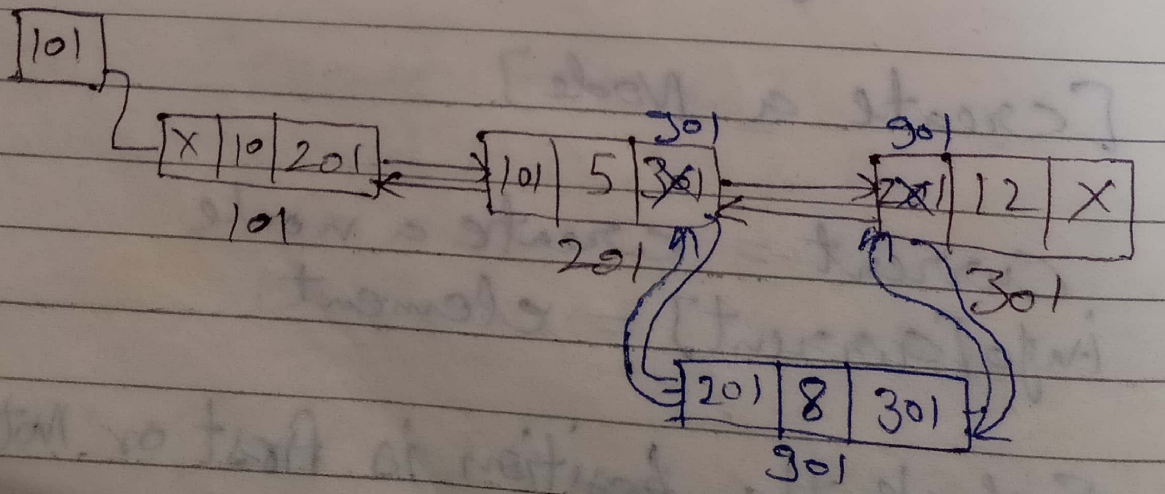
Step 4: I = 1, temp = start

Step 5: While I < pos - 1
temp = link[temp]
I = I + 1
Endwhile

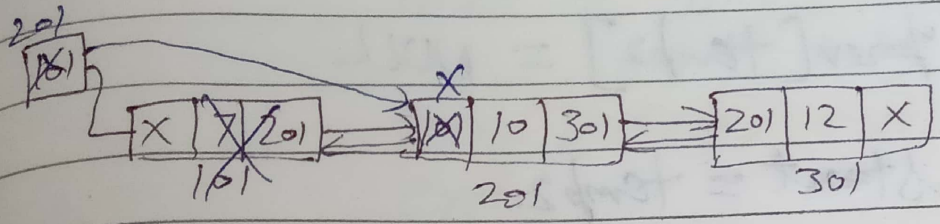
Step 6: temp2 = link[temp]

prev[current] = temp
link[temp] = current
link[current] = temp2
prev[temp2] = current

Step 7: END.



Algorithm to delete a Node from Beginning:-



delete_begining(start)

Step 1: [check the list is Empty or Not]

If (start = NULL)

print "List is empty"
exit

Endif.

Step 2: [check the List contains only one Node]

If link[start] = NULL

temp = start
start = link[temp]
OR NULL

free(temp)
exit

Endif.

Step 3: temp = start

Step 4: temp2 = link[temp]

Step 5: prev[temp2] = NULL

Step 6: start = temp2

Step 7: free (temp)

Step 8: END.

Algorithm to Delete a Node from End:-

delete_end (start)

Step 1: [check the list is empty or Not]

If start = NULL

print " List is empty"
exit
Endif.

Step 2: [check the List contains only one Node]

If link[start] = NULL

temp = start
start = link [temp]
OR NULL

free (temp)
exit

Endif

Step 3: temp = start

Step 4: While link [temp] ≠ NULL

temp = link [temp]

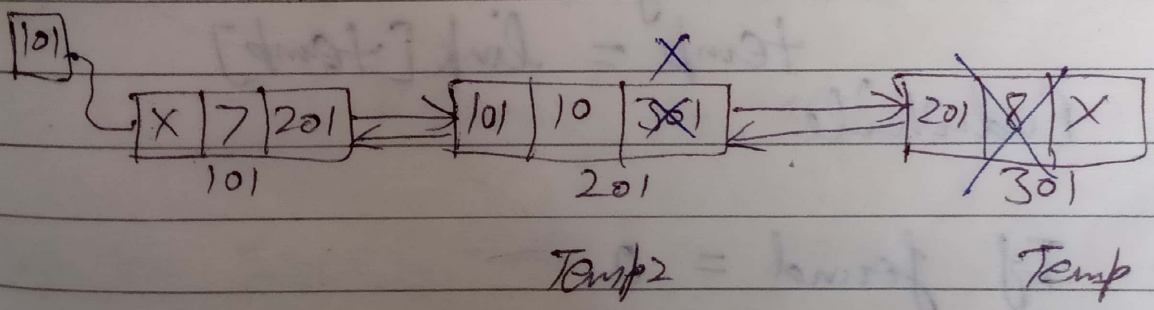
Endwhile

Step 5: temp2 = prev [temp]

Step 6: link [temp2] = NULL

Step 7: free (temp)

Step 8: End.



Algorithm to Delete Desired Element:-

delete_desired (start, element).

Step 1: [check the list is empty or Not]

```
If start = NULL
        print "List is Empty"
        exit
Endif.
```

Step 2: temp = start, found = 0

```
While temp ≠ NULL
        If info[temp] = element
                found = 1
                exit loop
        endif
        temp = link[temp]
EndWhile.
```

Step 3: If found = 0

```
        print " Not Found "
        exit
endif.
```

Step 4: If link[temp] = NULL

```
temp2 = prev[temp]
link[temp2] = NULL
free[temp]
exit
```

endif.

Step 5: If prev[temp] = NULL

```
temp2 = link[temp]
prev[temp2] = NULL
start = temp2
free(temp)
exit
```

Endif

Step 6: temp2 = link[temp]

Step 7: temp3 = prev[temp]

Step 8: link[temp3] = temp2

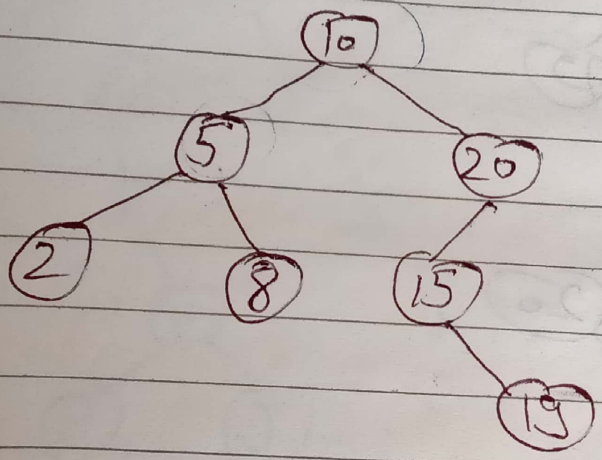
Step 9: prev[temp2] = temp3

Step 10: free(temp)

Step 11: END.

Binary Search Tree (BST):-

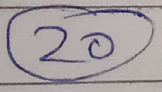
BST is a special binary tree whose each parent node must be greater than left child node and less than right child node.



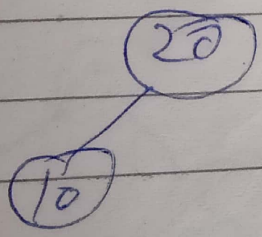
Insert the Element in BST:-

20 10 5 19 3 15 11

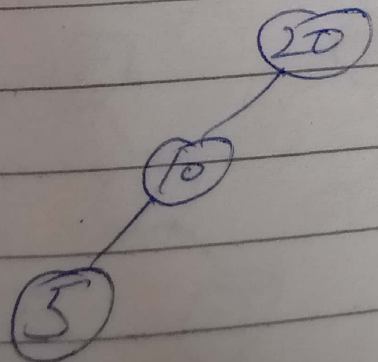
Insert 20



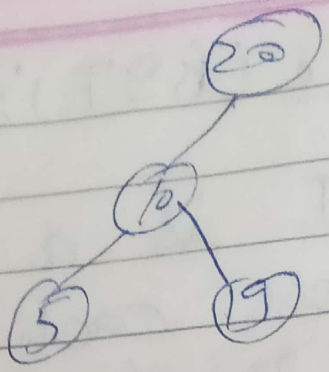
Insert 10



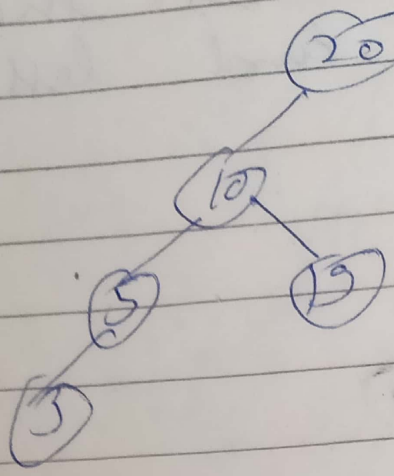
Insert 5



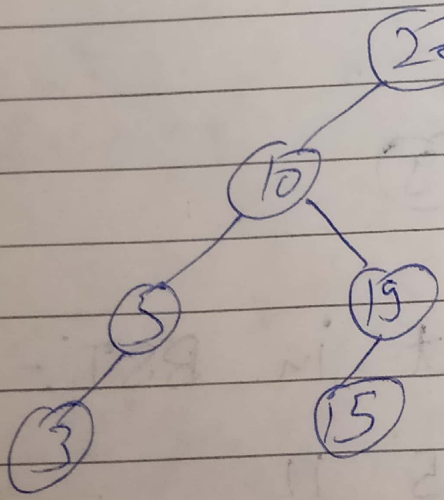
19



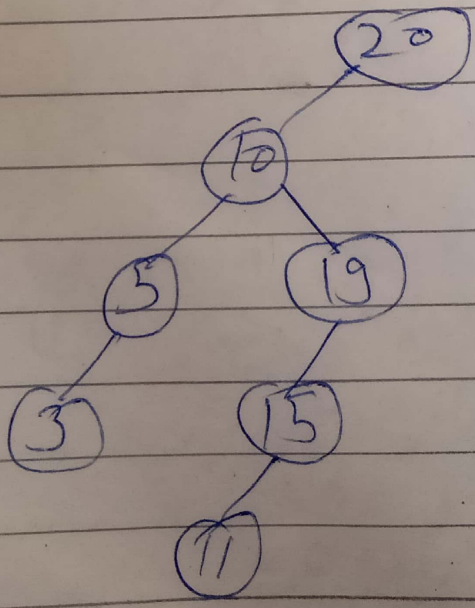
3.



15



11

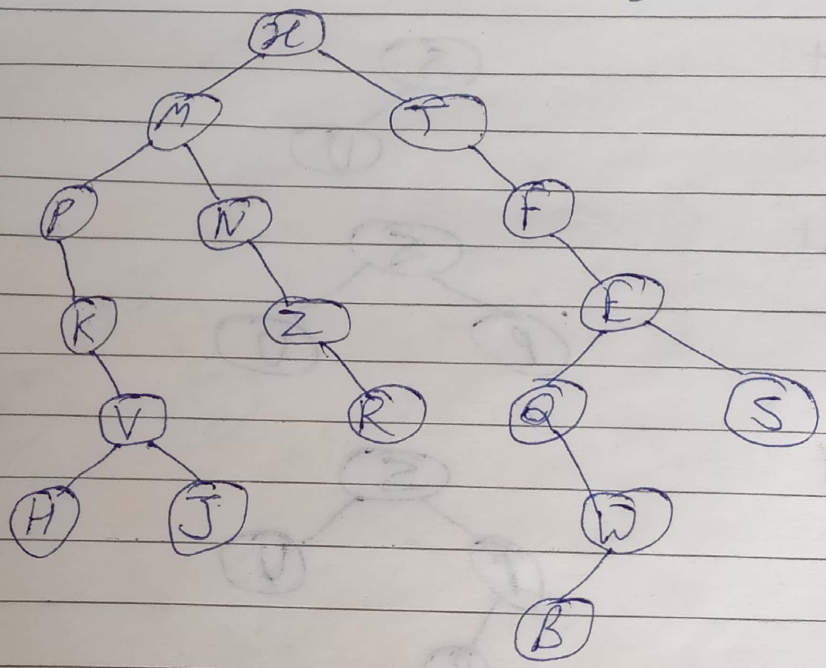


Ans-7- of Page No. 1 (Tree)

(i)

Pre- X M P K V H J N Z R T F E Q W B S

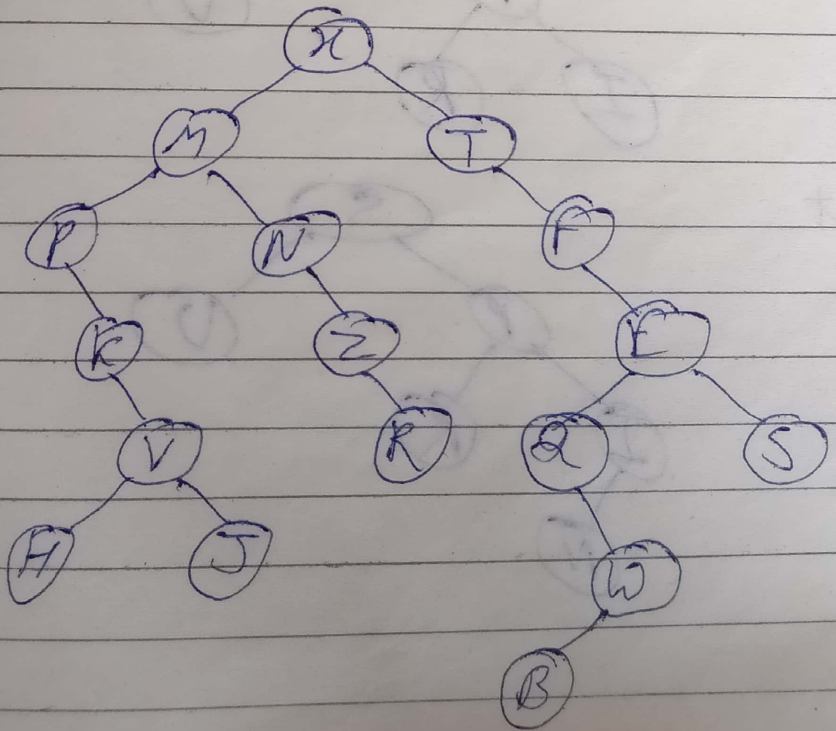
In- P K H V J M N Z R X T F Q B W E S



(ii)

Post- H J V K P R Z N M B W Q S E F T X

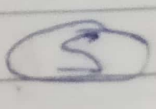
In- P K H V J M N Z R X T F Q B W E S



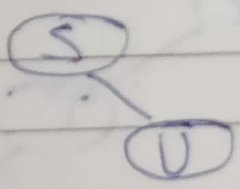
Create a BST for the following elements

S U P R I N T E D

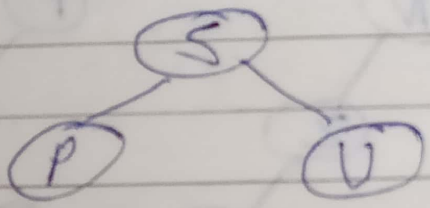
Insert S



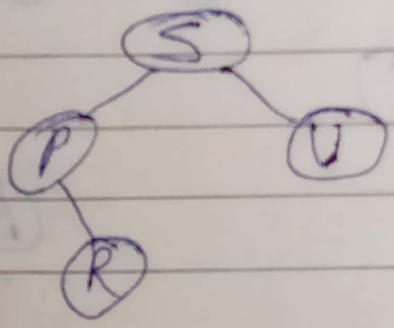
Insert U



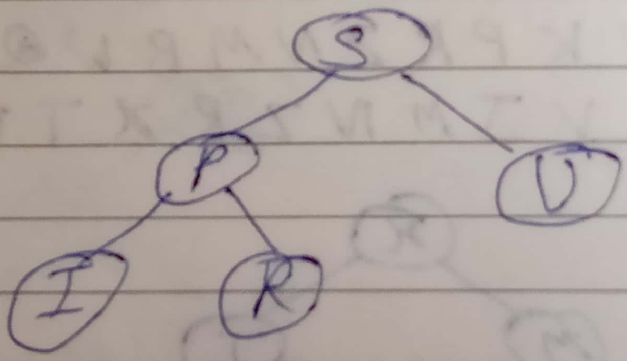
Insert P



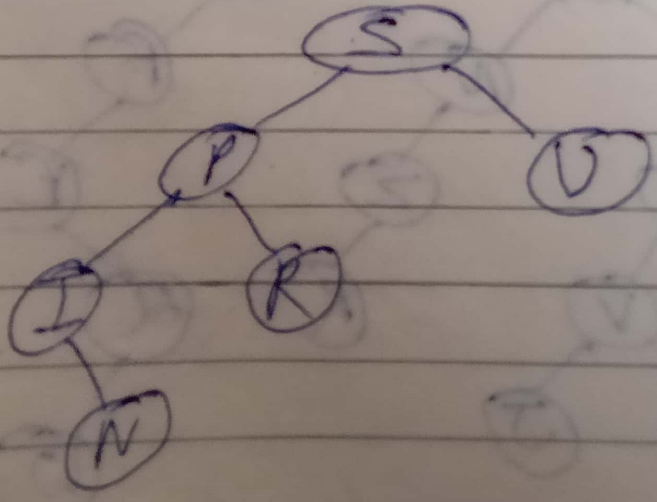
Insert R



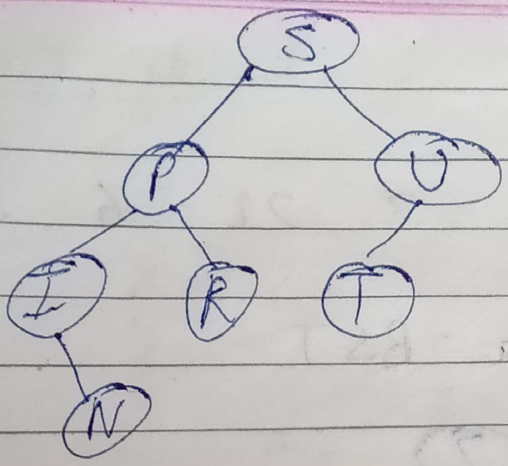
Insert I



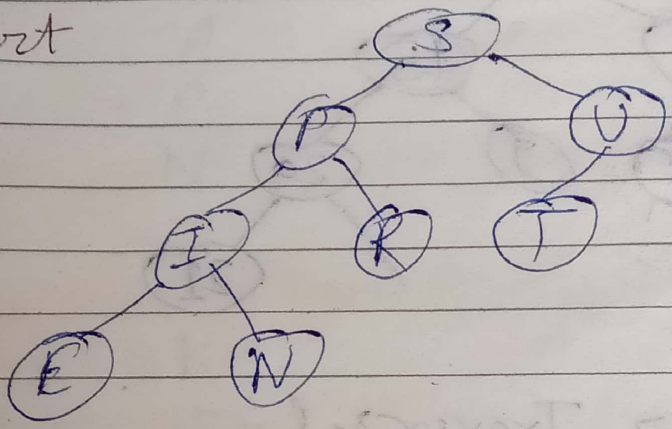
Insert N



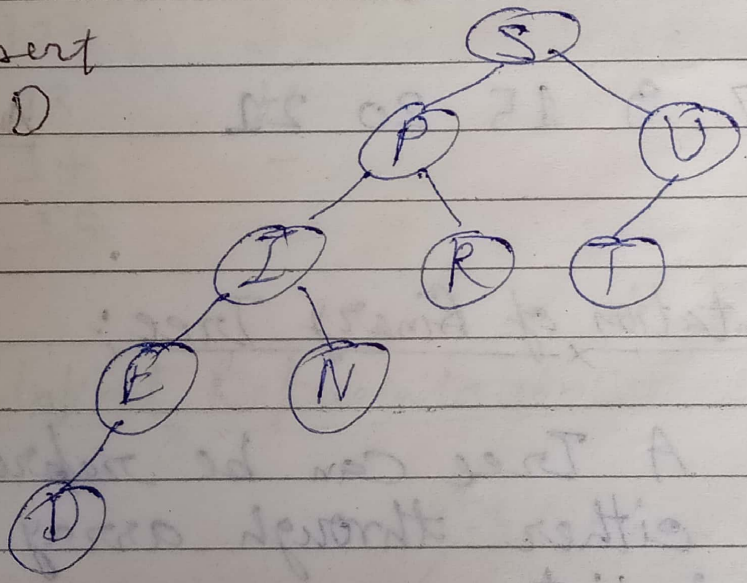
Insert
T



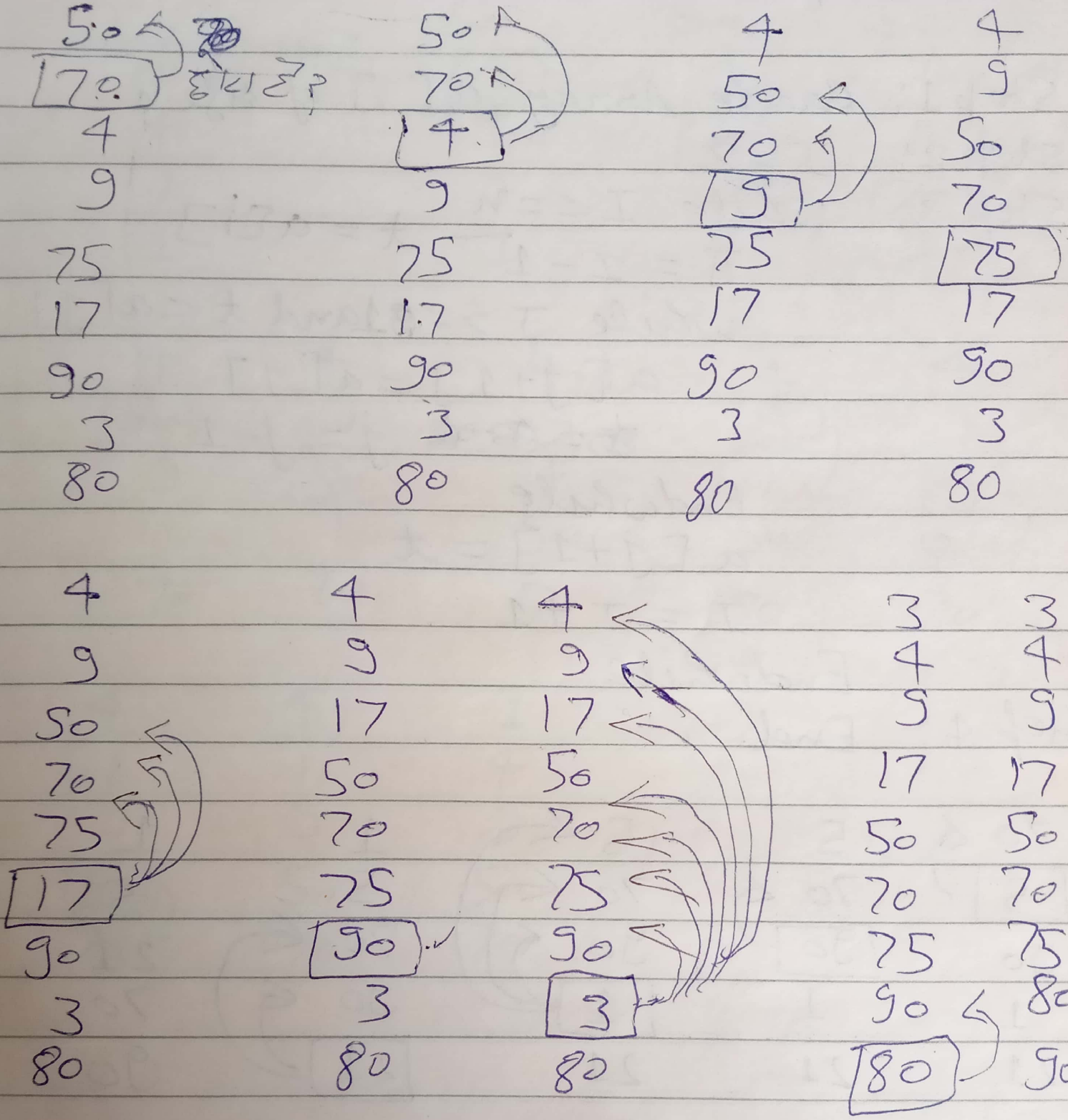
Insert
E



Insert
D

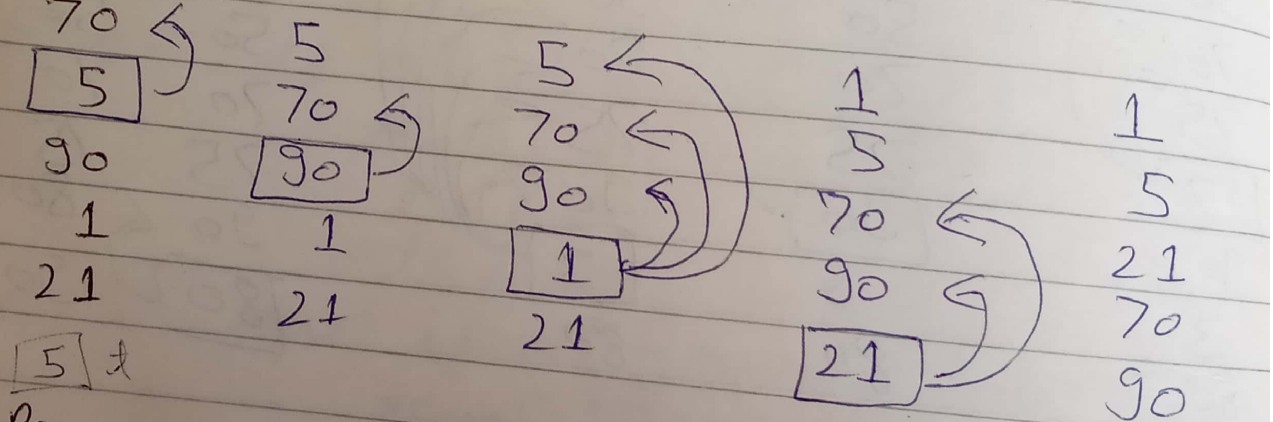


Insertion Sort :-



Algorithm for Insertion Sort:-

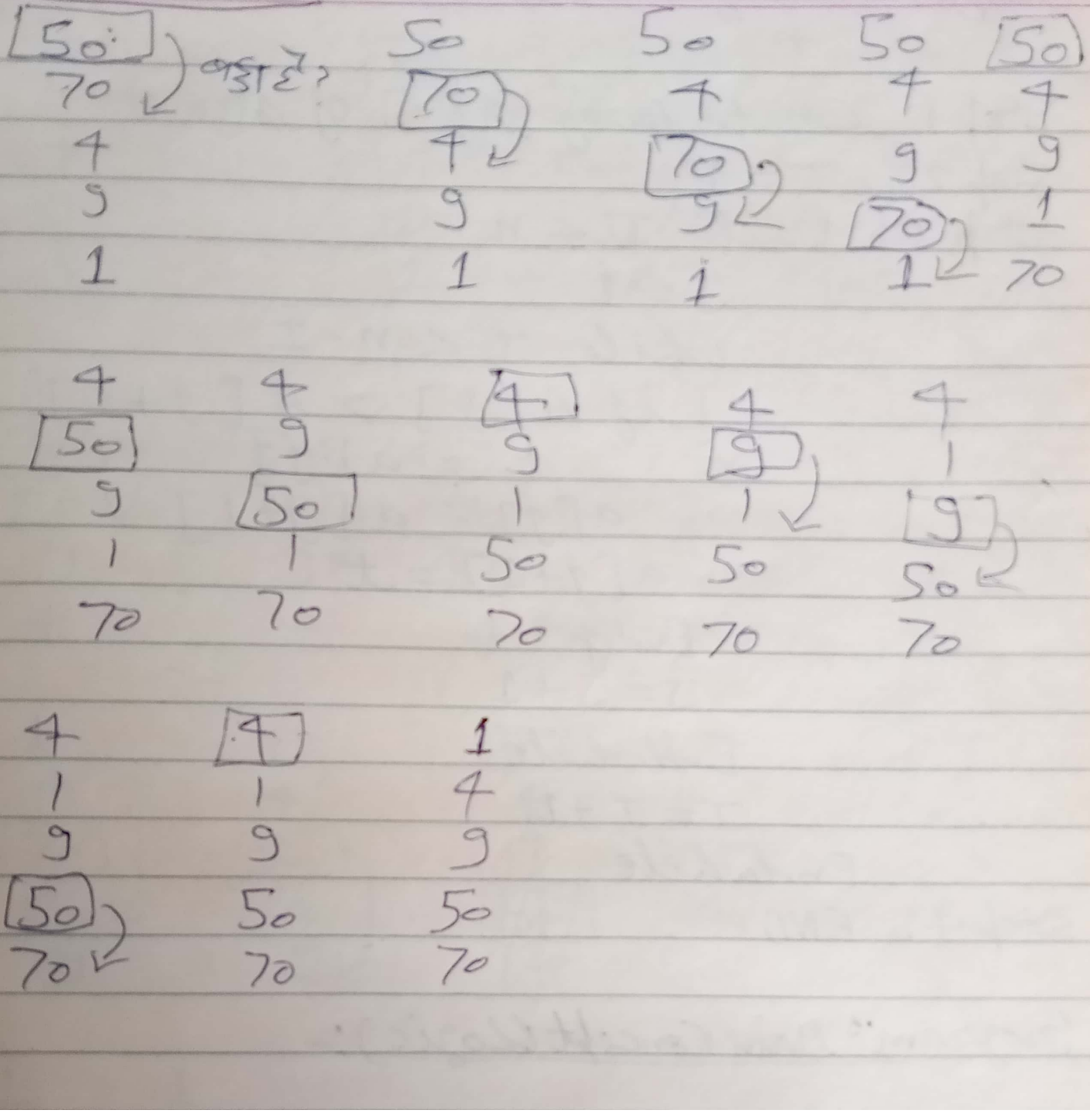
Step 1: Create Array a[ ] of size n.
step 2: I = 2
Step 3: While I <= n
t = a[i]
J = I - 1
while J >= 0 and t < a[j]
a[j+1] = a[j]
j = j - 1
Endwhile
a[j+1] = t
I = I + 1
Endwhile
Step 4: End.



Program's Main Concept (Logic):-

```
for(i=1; i<n; i++)
{
    t = a[i];
    for(j=i-1; j>=0 && t<a[j]; j--)
        a[j+1] = a[j];
    a[j+1] = t;
}
```

Bubble Sort:-


50
70
बड़ा है?
4
9
1
50
70
4
9
1
50
4
70
9
1
50
4
9
70
1
50
4
9
1
70
4
50
9
1
70
4
9
50
1
70
4
9
1
50
70
4
1
9
50
70
4
1
9
50
70
1
4
9
50
70

Algorithm for Bubble Sort:-

```
Step 1: Create Array A[ ] of size N
Step 2: I = 1
step 3: While I < n
            J = 1
            While J < n - I
              if A[J] > A[j+1]
                  t = a[j]
                a[j] = a[j+1]
              a[j+1] = t
            Endif
            J = J + 1
          Endwhile
          I = I + 1
        Endwhile
step 4: END
```

Program's Main Concept (Logic):-

```
for (i=0; i<n-1; i++)
    for (j=0; j<n-1-i; j++)
        if (a[j] > a[j+1])
        {
            t = a[j];
          a[j] = a[j+1];
        a[j+1] = t;
    }
```

t-Test

This t-distribution is used when sample size is ≤ 30 & the population standard deviation is unknown.

t-statistic is defined as

$$t = \frac{\bar{x} - \mu}{s/\sqrt{n}}$$

degree of freedom (d.f.) $= n-1$

$$s = \sqrt{\frac{\Sigma(X-\bar{X})^2}{n-1}}$$

Ex.(1.) Page 553 (Navish) To test of mean of a Random sample

$n = 16$, $\bar{X} = 53$, $\Sigma(X-\bar{X})^2 = 135$, $\mu = 56$

$$s = \sqrt{\frac{\Sigma(X-\bar{X})^2}{n-1}} \Rightarrow \sqrt{\frac{135}{16-1}} \Rightarrow \sqrt{\frac{135}{15}} \Rightarrow \sqrt{9}$$

$$s \Rightarrow 3$$

$$t = \frac{53-56}{3/\sqrt{16}} \Rightarrow \frac{-3}{3/4} \Rightarrow \frac{-12}{3} \Rightarrow -4$$

$|t| = 4$

d.f. $= n-1 \Rightarrow 16-1 \Rightarrow 15$

So, $t_{0.05} = 2.13 < |t|$

$= 2.13 < 4$

So, Hypothesis is rejected because calculated value of t is more than table value.

3+2

correct.

95% confidence limits of population mean $= \bar{x} \pm \frac{s}{\sqrt{n}} t_{0.05}$

$\Rightarrow 53 \pm \frac{3}{\sqrt{16}}(2.13) \Rightarrow 53 \pm \frac{3}{4}(2.13) \Rightarrow 53 \pm 0.75(2.13)$

$\Rightarrow 53 \pm 1.5975$

$\Rightarrow 54.5975, 51.4025$

Correct ✓ उत्तर.

99% confidence limits of population mean

$= \bar{x} \pm \frac{s}{\sqrt{n}} t_{0.01}$

$\Rightarrow 53 \pm \frac{3}{\sqrt{16}}(2.95) \Rightarrow 53 \pm \frac{3}{4}(2.95)$

$\Rightarrow 53 \pm 0.75(2.95) \Rightarrow 53 \pm 2.2125$

$\Rightarrow 55.2125, 50.7875$

Correct ✓ उत्तर.

Ex-2. Page 554 (Manish)

$n = 10, \mu = 4, \bar{x} = \frac{44}{10} \Rightarrow 4.4$

x	4.2	4.6	3.9	4.1	5.2	3.8	3.9	4.3	4.4	5.6
$x-\bar{x}$	-0.2	0.2	-0.5	-0.3	0.8	-0.6	-0.5	-0.1	0	1.2
$(x-\bar{x})^2$	0.04	0.04	0.25	0.09	0.64	0.36	0.25	0.01	0	1.44

$\Sigma(x-\bar{x})^2 = 3.12$

$s = \sqrt{\frac{\Sigma(x-\bar{x})^2}{n-1}} \Rightarrow \sqrt{\frac{3.12}{10-1}} \Rightarrow \sqrt{\frac{3.12}{9}}$

$\Rightarrow 0.588$

$t = \frac{\bar{x} - \mu}{s/\sqrt{n}} \Rightarrow \frac{4.4-4}{0.588/\sqrt{10}} \Rightarrow \frac{0.4}{0.588/3.16}$

$\Rightarrow \frac{0.4}{0.186} \Rightarrow 2.15$

So, $t_{0.05} = 2.26 > 2.15$

So, the hypothesis is accepted because calculated value of t is less than table value.

3-12 correct

Ex.-(3.) Page (554) (Punish)

$n = 20,\ \bar{x} = 42,\ S.D. = 5,\ \mu = 45$

$$s^2 = \frac{n}{n-1}(5)^2 \Rightarrow \frac{20}{20-1}(5)^2$$

$$s^2 \Rightarrow \frac{20}{19}25 \Rightarrow 26.315$$

$$s = 5.129$$

$$t = \frac{\bar{x} - \mu}{s/\sqrt{n}} \Rightarrow \frac{42-45}{5.129/\sqrt{20}} \Rightarrow \frac{-3}{5.129/4.47}$$

$$\Rightarrow \frac{-3}{1.147} \Rightarrow -2.615$$

$$|t| = 2.615$$

So, $t_{0.05} = 2.09 < 2.615$

So, Hypothesis is rejected because the calculated value is greater than the table value. ✓ Correct.

उत्तर.

Ex.-(3.) Page (554) (Manish)

$n = 20,\ \bar{x} = 42,\ S.D. = 5,\ \mu = 45$

$$s^2 = \frac{n}{n-1}(5)^2 \Rightarrow \frac{20}{20-1}(5)^2$$

$$s^2 \Rightarrow \frac{20}{19}25 \Rightarrow 26.315$$

$$s = 5.129$$

$$t = \frac{\bar{x} - \mu}{s/\sqrt{n}} \Rightarrow \frac{42-45}{5.129/\sqrt{20}} \Rightarrow \frac{-3}{5.129/4.47}$$

$$\Rightarrow \frac{-3}{1.147} \Rightarrow -2.615$$

$$|t| = 2.615$$

So, $t_{0.05} = 2.09 < 2.615$

So, Hypothesis is rejected because the calculated value is greater than the table value. ✓ Correct.

Ex-(4.) Page(555)(Manish)

$n = 9$, assumed mean $(\mu) = 47.5$,

$\bar{x} = \frac{442}{9} \Rightarrow 49.11$

x	45	47	50	52	48	47	49	53	51
$x-\bar{x}$	-4.11	-2.11	0.89	2.89	-1.11	-2.11	-0.11	3.89	1.89
$(x-\bar{x})^2$	16.89	4.45	0.79	8.35	1.23	4.45	0.0121	15.13	3.57

$\Sigma(x-\bar{x})^2 = 54.87$

$s = \sqrt{\frac{\Sigma(x-\bar{x})^2}{n-1}} \Rightarrow \sqrt{\frac{54.87}{9-1}} \Rightarrow \sqrt{\frac{54.87}{8}}$

$\Rightarrow \sqrt{6.85875} \Rightarrow 2.6189$

$t = \frac{\bar{x}-\mu}{s/\sqrt{n}} \Rightarrow \frac{49.11-47.5}{2.6189/\sqrt{9}} \Rightarrow \frac{1.61}{2.6189/3}$

$\Rightarrow \frac{1.61}{0.8729} \Rightarrow 1.84$

So, $t_{0.05} = 2.31 > 1.84$

So, the hypothesis is accepted because the value of t in table is greater than calculated value of t. ✓ correct

उत्तर.

ToS of means of 2 small samples (from a normal population)

Ex.-(1.)
Page(556)(Jnanish)

$n_1 = 8,\ n_2 = 7,\ S_1 = 36,\ S_2 = 40,$
$\bar{X}_1 = 1234,\ \bar{X}_2 = 1036$

$$s^2 = \frac{n_1(S_1)^2 + n_2(S_2)^2}{n_1 + n_2 - 2} \Rightarrow \frac{8(36)^2 + 7(40)^2}{8 + 7 - 2}$$

$$\Rightarrow \frac{8 \times 1296 + 7 \times 1600}{15 - 2} \Rightarrow \frac{10368 + 11200}{13}$$

$$s^2 \Rightarrow \frac{21568}{13} \Rightarrow 1659.076$$

$$s \Rightarrow 40.7317$$

$$t = \frac{\bar{X}_1 - \bar{X}_2}{s\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \Rightarrow \frac{1234 - 1036}{40.7317\sqrt{\frac{1}{8} + \frac{1}{7}}}$$

$$\Rightarrow \frac{198}{40.7317\sqrt{\frac{15}{56}}} \Rightarrow \frac{198}{40.7317\sqrt{0.2678}} \Rightarrow \frac{198}{40.7317 \times 0.5175}$$

$$\Rightarrow \frac{198}{21.0786} \Rightarrow 9.3934 \text{ Ans.}$$

So, $t_{0.05} = 2.16 < 9.3934$ for $(n_1 + n_2 - 2)$ d.f.

∴ Hypothesis is rejected as tabular value is smaller than calculated value. Ans.

✓ correct.

Ex-(2.) Page(557) (Manish)

$\bar{X}_1 = 20.3,\ \bar{X}_2 = 18.6,\ n_1 = 10,\ n_2 = 14,$
$s_1 = 3.5,\ s_2 = 5.2$

$$s^2 = \frac{n_1 s_1^2 + n_2 s_2^2}{n_1 + n_2 - 2} \Rightarrow \frac{10(3.5)^2 + 14(5.2)^2}{10 + 14 - 2}$$

$$\Rightarrow \frac{10 \times 12.25 + 14 \times 27.04}{24 - 2} \Rightarrow \frac{122.5 + 378.56}{22}$$

$$s^2 \Rightarrow \frac{501.06}{22} \Rightarrow 22.775$$

$$s \Rightarrow 4.772$$

$$t = \frac{\bar{X}_1 - \bar{X}_2}{s\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \Rightarrow \frac{20.3 - 18.6}{4.772\sqrt{\frac{1}{10} + \frac{1}{14}}}$$

$$\Rightarrow \frac{1.7}{4.772\sqrt{\frac{126}{70 \cdot 35}}} \Rightarrow \frac{1.7}{4.772\sqrt{0.1714}}$$

$$\Rightarrow \frac{1.7}{4.772 \times 0.414} \Rightarrow \frac{1.7}{1.9756} \Rightarrow 0.8604$$

So, $t_{0.05} = 2.07 > 0.8604$ for $(n_1 + n_2 - 2)$ degree of freedom.

∴ Hypothesis is accepted because calculated value is smaller than tabular value. Hence.

correct

F- Test

Ex-(1.) (manish)
Page(562)

$\bar{X}_1 = 21.625, \bar{X}_2 = 18.875, n_1 = 8, n_2 = 7$

X_1	$X_1 - \bar{X}_1$	$(X_1 - \bar{X}_1)^2$	X_2	$X_2 - \bar{X}_2$	$(X_2 - \bar{X}_2)^2$
17	-4.625	21.39	16	-2.875	8.26
27	5.375	28.89	16	-2.875	8.26
18	-3.625	13.14	20	1.125	1.26
25	3.375	11.39	27	8.125	66.02
27	5.375	28.89	26	7.125	50.77
29	7.375	54.39	25	6.125	37.52
13	-8.625	74.39	21	2.125	4.52
17	-4.625	21.39	—	—	—

$$s_1^2 = \frac{\Sigma(x_1 - \bar{x}_1)^2}{n_1 - 1} \Rightarrow \frac{253.87}{8-1} \Rightarrow \frac{253.87}{7}$$

$$\Rightarrow 36.267$$

$$s_2^2 = \frac{\Sigma(x_2 - \bar{x}_2)^2}{n_2 - 1} \Rightarrow \frac{176.61}{7-1} \Rightarrow \frac{176.61}{6}$$

$$\Rightarrow 29.435$$

$$F = \frac{s_1^2}{s_2^2} \Rightarrow \frac{36.267}{29.435} \Rightarrow 1.23$$

So, $F_{0.05} = 4.21 > 1.23$

∴ Hypothesis is accepted because calculated value is less than tabulated

(2) value. ✓ (correct.)

Ex. (2) Page (563) (Manish)

$\bar{x}_1 = 31, \bar{x}_2 = 28, n_1 = 7, n_2 = 6$

x_1	$x_1 - \bar{x}_1$	$(x_1 - \bar{x}_1)^2$	x_2	$x_2 - \bar{x}_2$	$(x_2 - \bar{x}_2)^2$
28	−3	9	29	1	1
30	−1	1	30	2	4
32	1	1	30	2	4
33	2	4	24	−4	16
31	0	0	27	−1	1
29	−2	4	28	0	0
34	3	9	—	—	—

$$s_1^2 = \frac{\Sigma (x_1 - \bar{x}_1)^2}{n_1 - 1} \Rightarrow \frac{28}{7-1} \Rightarrow \frac{28}{6} \Rightarrow 4.67$$

$$s_2^2 = \frac{\Sigma (x_2 - \bar{x}_2)^2}{n_2 - 1} \Rightarrow \frac{26}{6-1} \Rightarrow \frac{26}{5} \Rightarrow 5.2$$

$$F = \frac{s_2^2}{s_1^2} \Rightarrow \frac{5.2}{4.67} \Rightarrow 1.1135$$

So, $F_{0.05} = 4.35 > 1.1135$ (3)

∴ Hypothesis is accepted because the tabular value is greater than the calculated value. ✓ (correct)

~~Ex.-(3.) Page (564)~~ (Manish)

$n_1 = 16,\ n_2 = 25,\ s_1^2 = 40,\ s_2^2 = 42$

$$F = \frac{s_1^2}{s_2^2}$$

as, we know that $s_1^2 = \frac{n_1 (S_1)^2}{n_1 - 1}$

$$s_2^2 = \frac{n_2 (S_2)^2}{n_2 - 1}$$

$$F = \frac{\frac{n_2 (S_2)^2}{n_2 - 1}}{\frac{n_1 (S_1)^2}{n_1 - 1}} \Rightarrow \frac{\frac{25(42)^2}{25-1}}{\frac{16(40)}{16-1}}$$

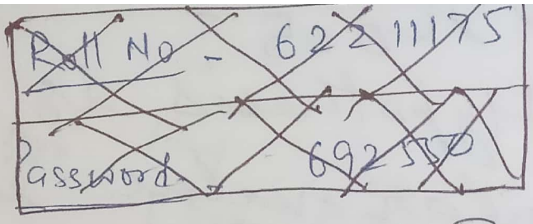
$$\Rightarrow \frac{\frac{25 \times 42}{24}}{\frac{16 \times 40}{15}} \Rightarrow \frac{5_{25 \times 42 \times 15}}{\frac{24}{8} \times \frac{16}{8} \times \frac{40}{8}}$$

$$\Rightarrow \frac{5 \times 15 \times 7}{8 \times 8 \times 8} \Rightarrow 1.025$$

So, $F_{0.05} = 2.29 > 1.025$

Here, calculated value is less than tabular value; So, hypothesis is accepted.

CHI-Square (χ^2) Test



Ex. (1.) Page (568) (Manish)

$$Ei = \frac{84}{6} \Rightarrow 14 \text{ (Expected Freq.)}$$

Observed freq. (Oi)	14	18	12	11	15	14
Expected Freq. (Ei)	14	14	14	14	14	14
$(Oi-Ei)^2$	0	16	4	9	1	0

$$\chi^2 = \frac{\Sigma (Oi-Ei)^2}{Ei} \Rightarrow \frac{30}{14} \Rightarrow 2.14\overline{28}$$

So, $\chi^2_{0.05} = 11.07 > 2.1428$

∴ Hypothesis is accepted because tab-ular value is greater than calculated value. ~~correct~~ ✓

Ex. (2.) Page (568) (manish)

Expected Freq. $(Ei) = \frac{276}{6} \Rightarrow 46$

Oi	40	32	29	59	57	59
Ei	46	46	46	46	46	46
$(Oi-Ei)^2$	36	196	289	169	121	169

$$\chi^2 = \frac{\Sigma (Oi-Ei)^2}{Ei} \Rightarrow \frac{980}{46}$$

$$\Rightarrow 21.3$$

② So, $\chi^2_{0.05} = 11.07 < 21.3$

∴ Hypothesis is rejected because tabular value is less than calculated value. ∴ ✓ (correct.)

Ex. (3.) (Manish)

Page (569)

Expected freq. $(E_i) = \frac{10000}{10}$

$= 1000$

O_i	1026	1107	997	966	1075	933	1107	972	964	853
E_i	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
$(O_i - E_i)^2$	676	11449	9	1156	5625	4489	11449	784	1296	21609

$$\chi^2 = \frac{\Sigma (O_i - E_i)^2}{E_i} \Rightarrow \frac{58542}{1000} \Rightarrow 58.542$$

So, $\chi^2_{0.05} = 16.919 < 58.542$

∴ Hypothesis is rejected because the tabular value is less than the calculated value. ∴ ✓ (correct.)

Linear Regression

σ = standard deviation

(1) regression coefficient b_{yx}. b_{xy}.

$r = \sqrt{b_{yx} \cdot b_{xy}}$

↓

correlation coefficient +.

$b_{yx} = r \frac{\sigma_y}{\sigma_x}$

$b_{xy} = r \frac{\sigma_x}{\sigma_y}$

(2) Linear regression

$$b_{yx} = \frac{n\Sigma xy - \Sigma x \Sigma y}{n\Sigma x^2 - (\Sigma x)^2}$$

$$b_{xy} = \frac{n\Sigma xy - \Sigma x \cdot \Sigma y}{n\Sigma y^2 - (\Sigma y)^2}$$

(3) Regression line y on x.

$$\boxed{y - \bar{y} = b_{yx}(x - \bar{x})}$$

$\bar{y} = \frac{\Sigma y}{n}$, $\bar{x} = \frac{\Sigma x}{n}$.

Regression line x on y.

$$\boxed{x - \bar{x} = b_{xy}(y - \bar{y})}$$

Non Linear Regression

$$y = a + bx + cx^2$$ (parabolic curve equation)

(Σ) $\Sigma y = na + b\Sigma x + c\Sigma x^2$

(Σx) $\Sigma xy = a\Sigma x + b\Sigma x^2 + c\Sigma x^3$

(Σx^2) $\Sigma x^2 y = a\Sigma x^2 + b\Sigma x^3 + c\Sigma x^4$

Straight line

$$\boxed{y = a + bx}$$

$$\Sigma y = na + b\Sigma x$$

$$\Sigma xy = a\Sigma x + b\Sigma x^2$$