Rajshahi University of Engineering & Technology
Department of Computer Science of Engineering

EXPERIMENT NO: 03
NAME OF EXPERIMENT: Arrays, Records and Pointers

SUBMITTED TO:

RIZOAN TOUFIQ
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
RAJSHAHI UNIVERSITY OF ENGINEERING & TECHNOLOGY

SUBMITTED BY:

NAME: MD. ARIFUL ISLAM
ROLL NO.: 1803046
GROUP: 2ND THIRTY
DATE OF EXP.: 16-09-2019
DATE OF SUB. : 23-09-2019
SERIES: 18 SERIES

MACHINE CONFIGURATION:

ASUS X510UF
CORE I5 8TH GEN PROCESSOR
UP TO 3.4 GHZ
8 GB RAM
OS WIN 10

INDEX

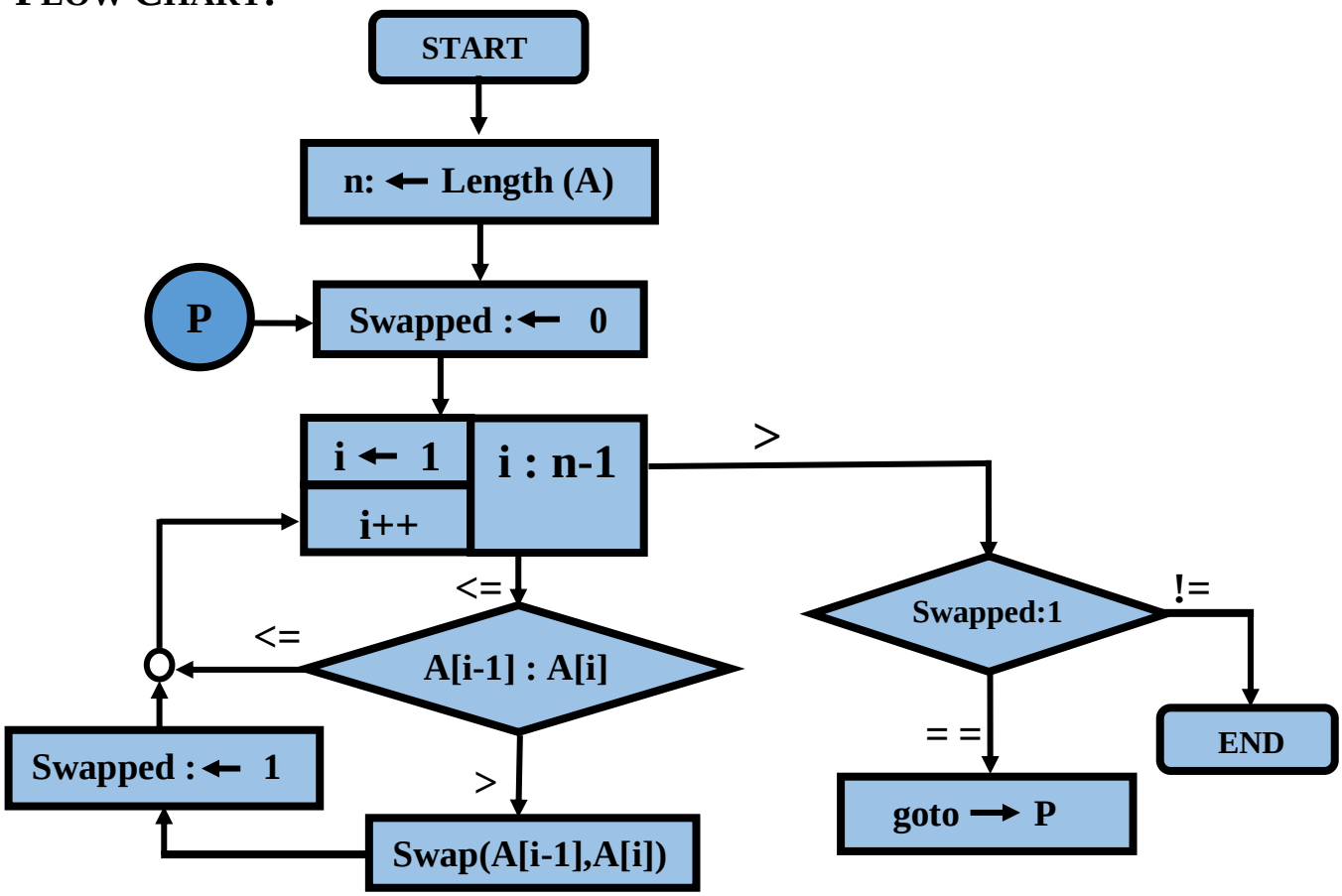
THEORY: A LINEAR ARRAY IS A LIST OF FINITE NUMBER N OF HOMOGENEOUS DATA ELEMENTS (I.E., DATA ELEMENTS OF THE SAME TYPE) SUCH THAT:
1.The elements of the array are referenced respectively by an Index set consisting of n consecutive numbers.
2. The elements of the **Array** are stored respectively in successive memory locations.
The number **n** of elements is called the **Length** or **Size** of the Array**.**

In the lab we -

1. Find the **Complexity of Bubble Sort**. In Bubble Sorting we can sort an array in Ascending or Descending order.
2. Searched an element from an array by **Linear Search**. In Linear Search we check all the elements of an array one by one until we find out our searched data.
3. Searched an element from an array by **Binary Search**. For binary search the **Array must be Sorted.** In Binary Search we check the elements of an array part by part by divideing the array in some condition.
4. Multiplied two matrixes (**Matrix Multiplication**).

PROBLEM 1: Find the Complexity of the bubble Sort algorithm.

FLOW CHART:



ALGORITHM:

(Given a nonempty array A with n numerical values**.** This algorithm sorts the values)

1. Set n := length(A)
2. swapped = false
3. Repeat i = 1 to n-1 by 1
4. If A[i-1]>A[i] then:
   Swap (A[i-1], A[i]) and swapped = true.
   [End of If Structure]
   [End of Repeat 4 loop]
5. If swapped = true then:
   Go to step 2
   [End of If structure]
6. Exit

CODE:

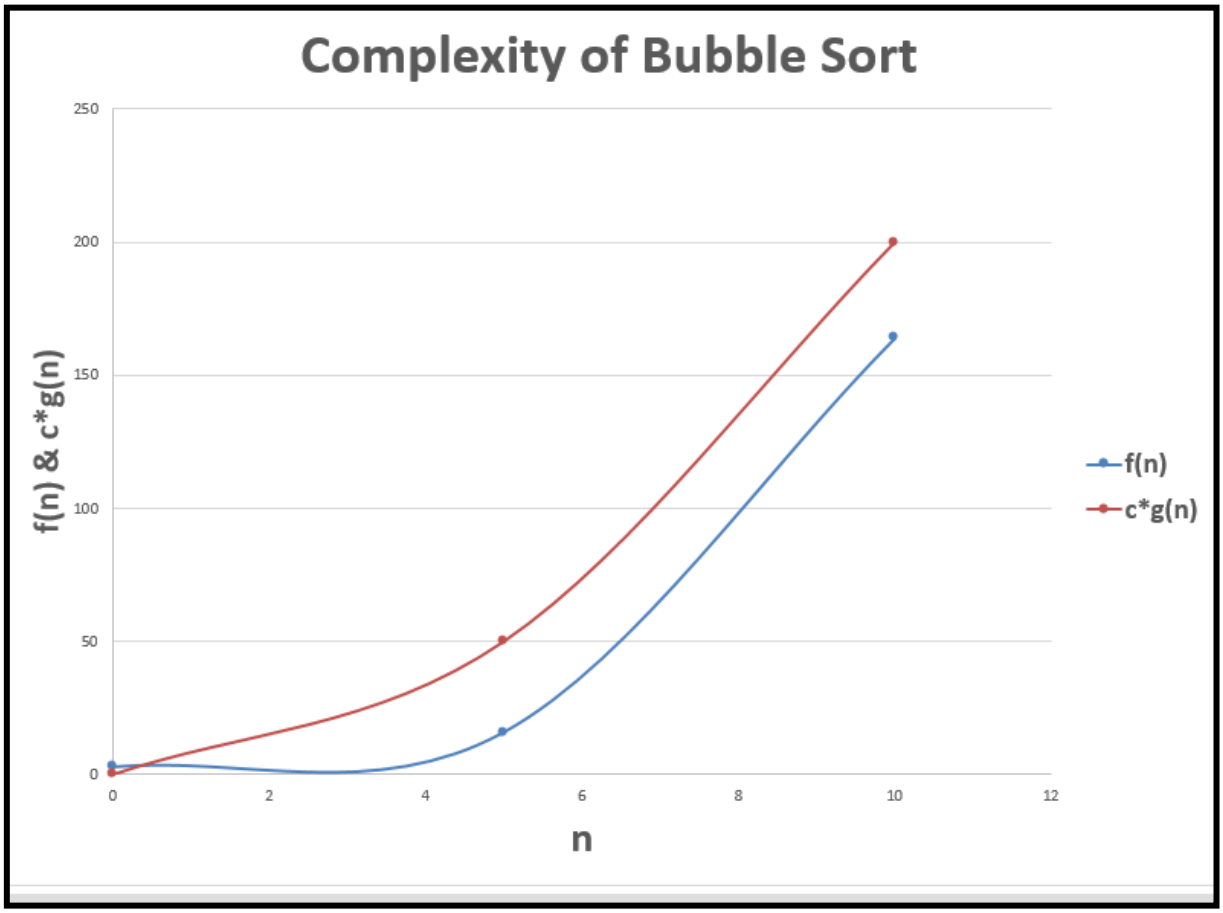
```
#include<stdio.h>

int main()
{
   int A[]={43,21,67,98,76,47,23,12,69,55},n,i,j=0,temp,swapped,p;
   n=sizeof(A)/4;
   p: swapped=0;
   for(i=1;i<n;i++)
   {
      if(A[i-1]>A[i])
      {
         temp=A[i-1];
         A[i-1]=A[i];
         A[i]=temp;
         swapped=1;
       }
    }

   if(swapped==1)
      goto p;

   return 0;
}
```

Complexity: The complexity of the above algorithm is O(n2). (Let c= 4)

Complexity Table:

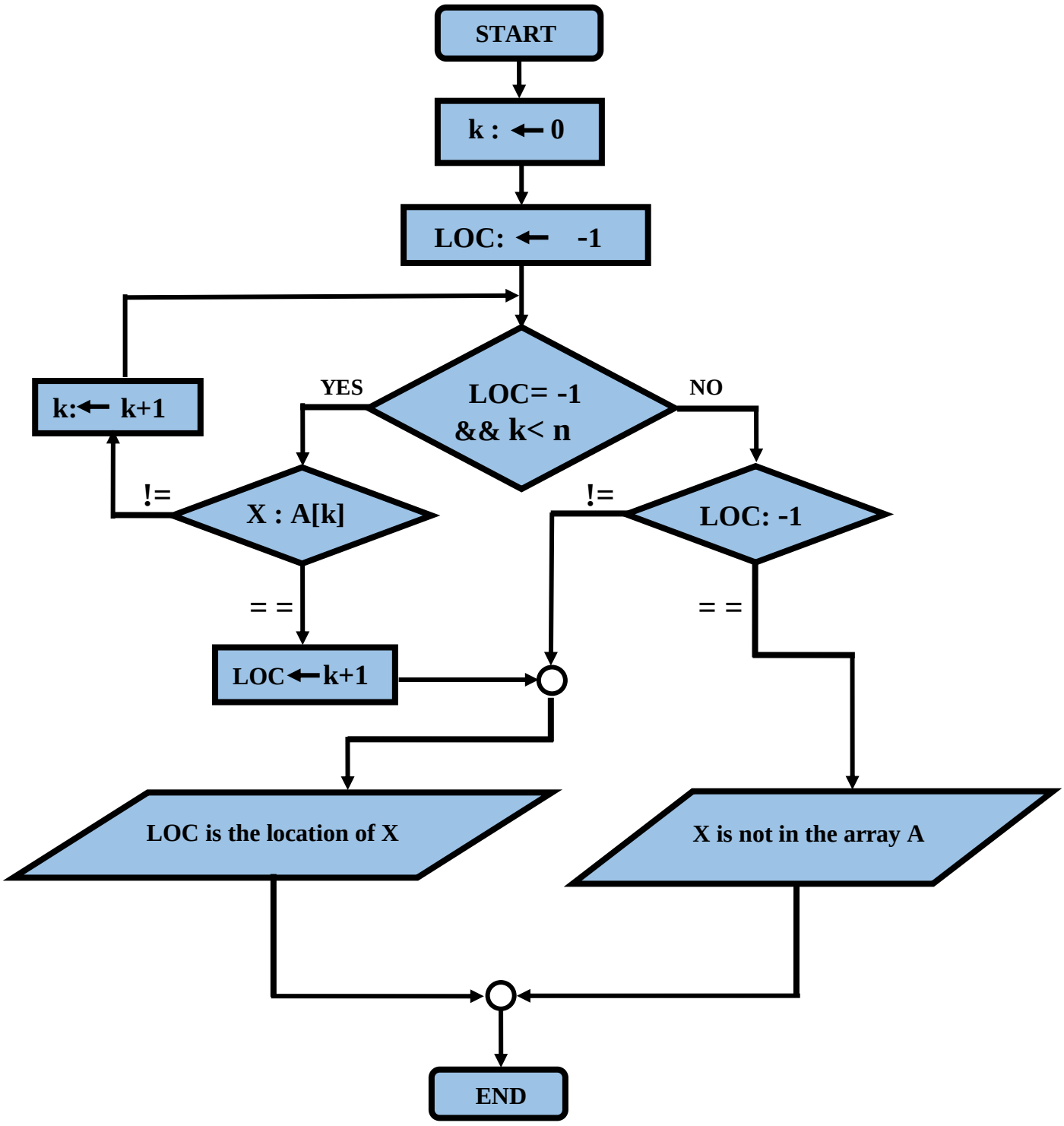
n	f(n)	c*g(n)
0	3	0
5	16	50
10	164	200

GRAPH:



PROBLEM 2: Linear Search from an Linear Array.

FLOW CHART:



ALGORITHM:

Given a nonempty array A with n numerical values and a specific x of information is given**.** This algorithm finds the location LOC of x in the array A or Sets LOC=-1)

1. Set K:=1, LOC:=-1
2. Repeat steps 3 and 4 while LOC = -1 and K≤n
3. IF x = A[K] then:
   Set LOC:=K.
   [End of If structure]
4. K:=K+1.
   [End of step 2 loop]
5. If LOC = -1 then: Write: x is not in the array A.
   Else: Write: LOC is the location of x
   [End of If structure]
6. Exit

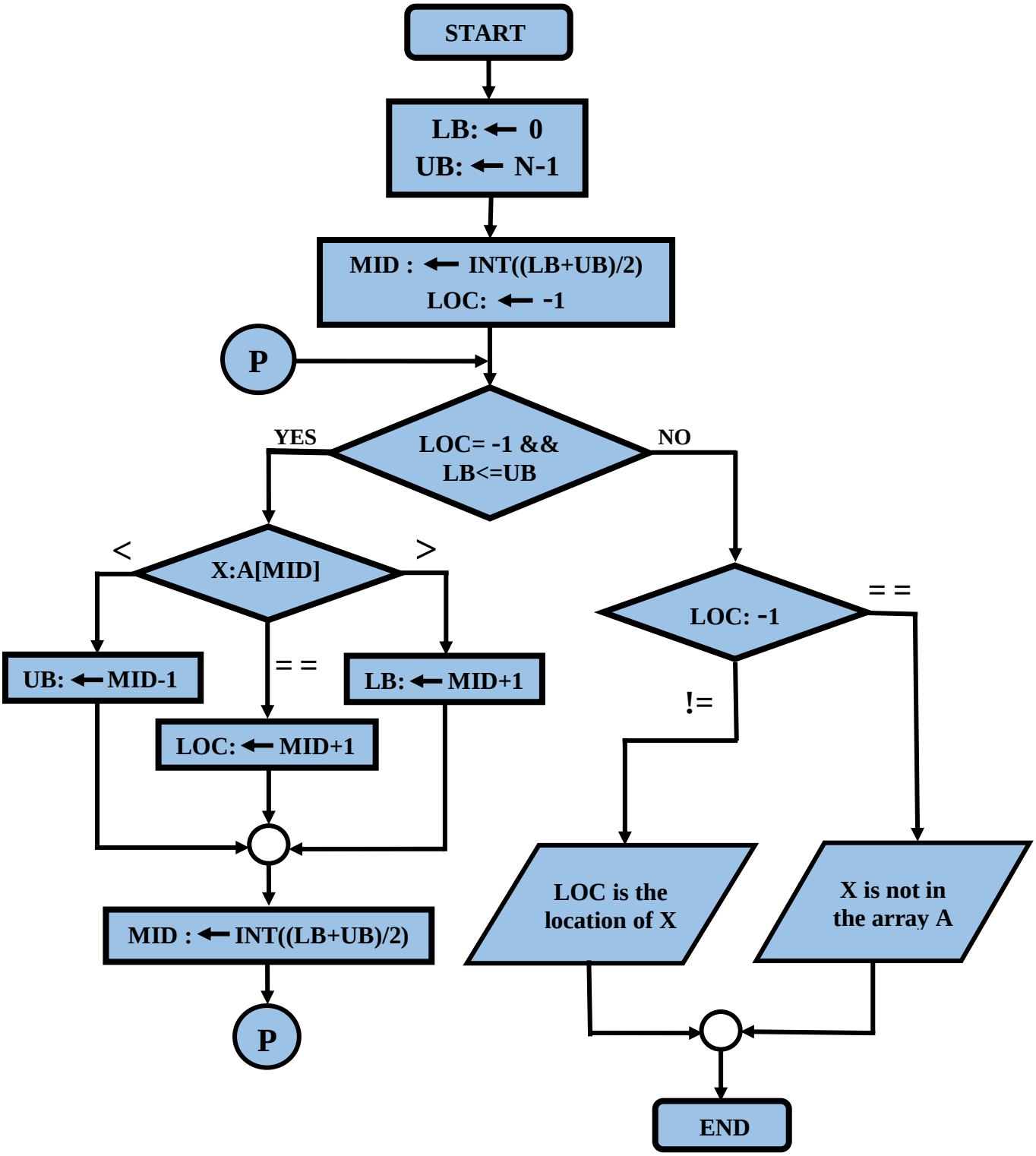
CODE:

```
#include<stdio.h>

int main()
{
   int A[]={43,21,67,98,76,47,23,12,69,55},x,k=0,n,loc=-1;
   n=sizeof(A)/4;
   scanf("%d",&x);
   while(loc==-1&&k<n)
   {
      if(x==A[k])
         loc=k+1;
      k++;
   }
   if(loc==-1)
      printf("%d is not in the Array\n",x);
   else
      printf("%d is the location of %d\n",loc,x);

   return 0;
}
```

Complexity: The worst case complexity is C(n) =n when X is the last element of the array and the average case complexity is C(n) = (n+1)/2.

PROBLEM 3: Binary Search from an Linear Array.

FLOW CHART:



ALGORITHM:

(Given a sorted array A with n numerical values and a specific x of information is given**.** This algorithm finds the location LOC of x in the array A or Sets LOC=-1)

1. Set LB:=1, UB:=n, MID = INT((LB+UP)/2) and LOC:=-1
2. Repeat steps 3 and 4 while LOC = -1 and LB≤UB
3. IF x < A[MID] then:
   Set UB:=MID-1.
   Else If x > A[MID] then:
   Set LB:=MID+1.
   Else:
   Set LOC:=MID
   [End of If structure]
4. MID = INT((LB+UP)/2).
   [End of step 2 loop]
5. If LOC = -1 then: Write: x is not in the array A.
   Else: Write: LOC is the location of x
   [End of If structure]
6. Exit

CODE:

```
#include<stdio.h>

 int main()
 {
    int A[]={23,26,31,37,45,53,65,69,74,78,81,85,89,93,99};
    int n,x,lb,ub,mid,loc=-1;

    n=sizeof(A)/4;
    lb=0;  ub=n-1;
    mid=(lb+ub)/2;
    scanf("%d",&x);
```

```
    while(loc==-1&&lb<=ub)
    {
       if(x<A[mid])
          ub=mid-1;
       else if(x>A[mid])
          lb=mid+1;
       else
          loc=mid+1;
      mid=(lb+ub)/2;
    }
    if(loc==-1)
      printf("%d is not in the Array\n",x);
    else
      printf("%d is the location of %d\n",loc,x);


    return 0;
}
```

Complexity:

After,

Loop 1 ---------- $n / 2^1$

Loop 2 ---------- $n / 2^2$

Loop 3 ---------- $n / 2^3$

.....

.....

After the last loop (say, k) there will be only one element in the arrey.

So after,

Loop k ---------- $1 = n / 2^k$

As, $1 = n / 2^k$

or, $2^k = n$

or, $\log_2 2^k = \log_2 n$

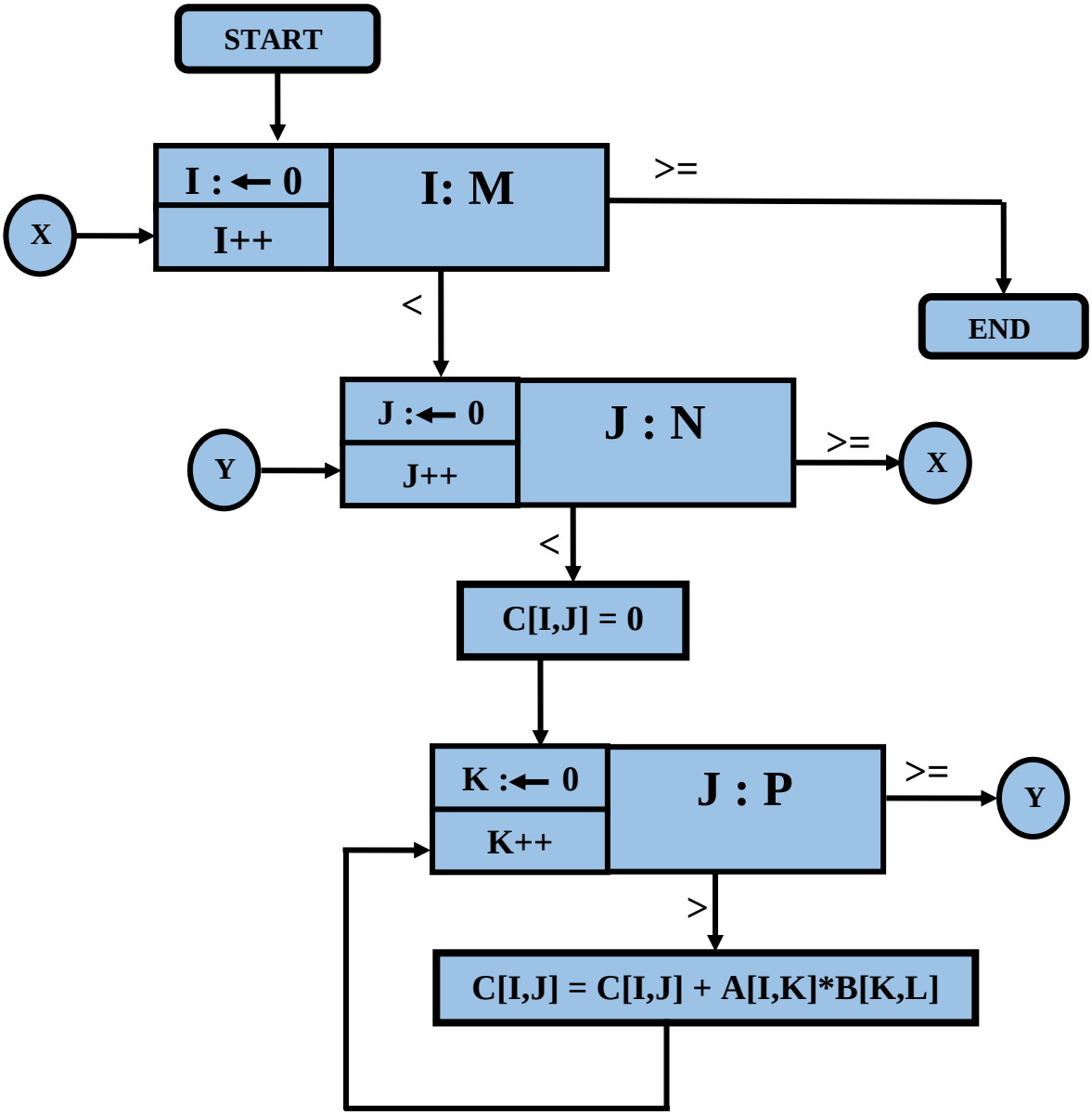
or, $k \log_2 2 = \log_2 n$

or, $k = \log_2 n$

So the complexity is **$O(\log_2 n)$.**

PROBLEM 4: Multiplication of Matrix.

FLOW CHART:



ALGORITHM:

(Let A be an M × P matrix array and B be an P × N matrix array. This algorithm stores the product of A and B in an M×N matrix array C)

1. Repeat steps 2 to 4 for I = 1 to M:
2. Repeat steps 3 and 4 for J = 1 to N:
3. Set C[I,J]:=0.
4. Repeat for K = 1 to P: C[I,J] :=C[I,J] + A[I,K]*B[K,J].

[End of step 4 loop]
[End of step 2 loop]
[End of step 1 loop]
5. Exit

CODE:

```
#include<stdio.h>

int main()
{
   int A[3][2]={2,3,1,2,5,3},B[2][3]={1,4,2,3,1,3};
   int C[3][3],i,j,k;

   for(i=0;i<3;i++)
      for(j=0;j<3;j++)
      {
         C[i][j]=0;
         for(k=0;k<2;k++)
            C[i][j]=C[i][j]+A[i][k]*B[k][j];
      }

   return 0;
}
```

Complexity: In the programe there is running 3 nested 'for' loop. We know each loop's complexity is n. So for 3 nested for loop the complexity will be $n*n*n = n^3$.

So the complexity is **$O(n^3)$.**

DISCUSSION: The problems were based on Array. After trying sometime I had solved the problems.

The End