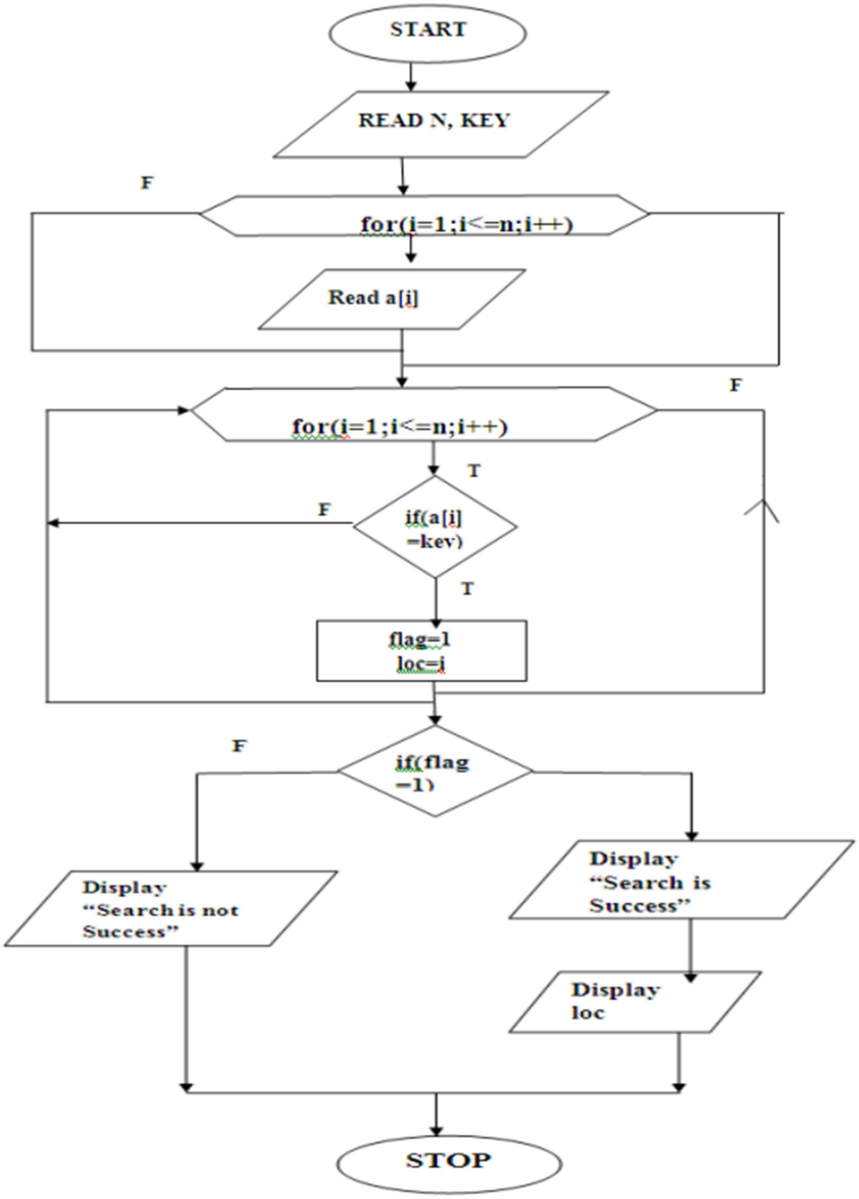
FLOWCHART


START
READ N, KEY
F
for(i=1;i<=n;i++)
Read a[i]
F
for(i=1;i<=n;i++)
T
F
if(a[i]
=key)
T
flag=1
loc=i
F
if(flag
=1)
Display
"Search is not
Success"
Display
"Search is
Success"
Display
loc
STOP

8. Write a c program that uses functions to perform the following:

i. Addition of two matrices.

ii. Multiplication of two matrices.

AIM: i) To find the Addition of Two Matrices using C program

ALGORITHM

Step1: Start

Step2: Input values are a[10][10],b[10[10]

Step3: Read rows and columns of first matrix as m,n

Step4: Read rows and columns of first matrix as p,q

Step5: initialize i=1,j=1

Step6: Check the condition if(i<=m) then go to step 7

Else goto step 9

Step7: Check the condition if(j<=n) then go to step 8

Else increment i and goto step 6

Step8: Read a[i][j] then increment j and goto step 7

Step9: Check the condition if(i<=p) then go to step 10

Else goto step 12

Step10: Check the condition if(j<=q) then go to step 11

Else increment i and goto step 9

Step11: Read b[i][j] then increment j and goto step 10

Step12: Check the condition if(m==p)&&(n==q) then go to step 13

Else goto step 18

Step13: Display matrix addition is possible and goto step 14

Step14: Check the condition if(i<=m) then go to step 15

Else goto step 17

Step15: Check the condition if(j<=n) then go to step 16

Else increment i and goto step 14

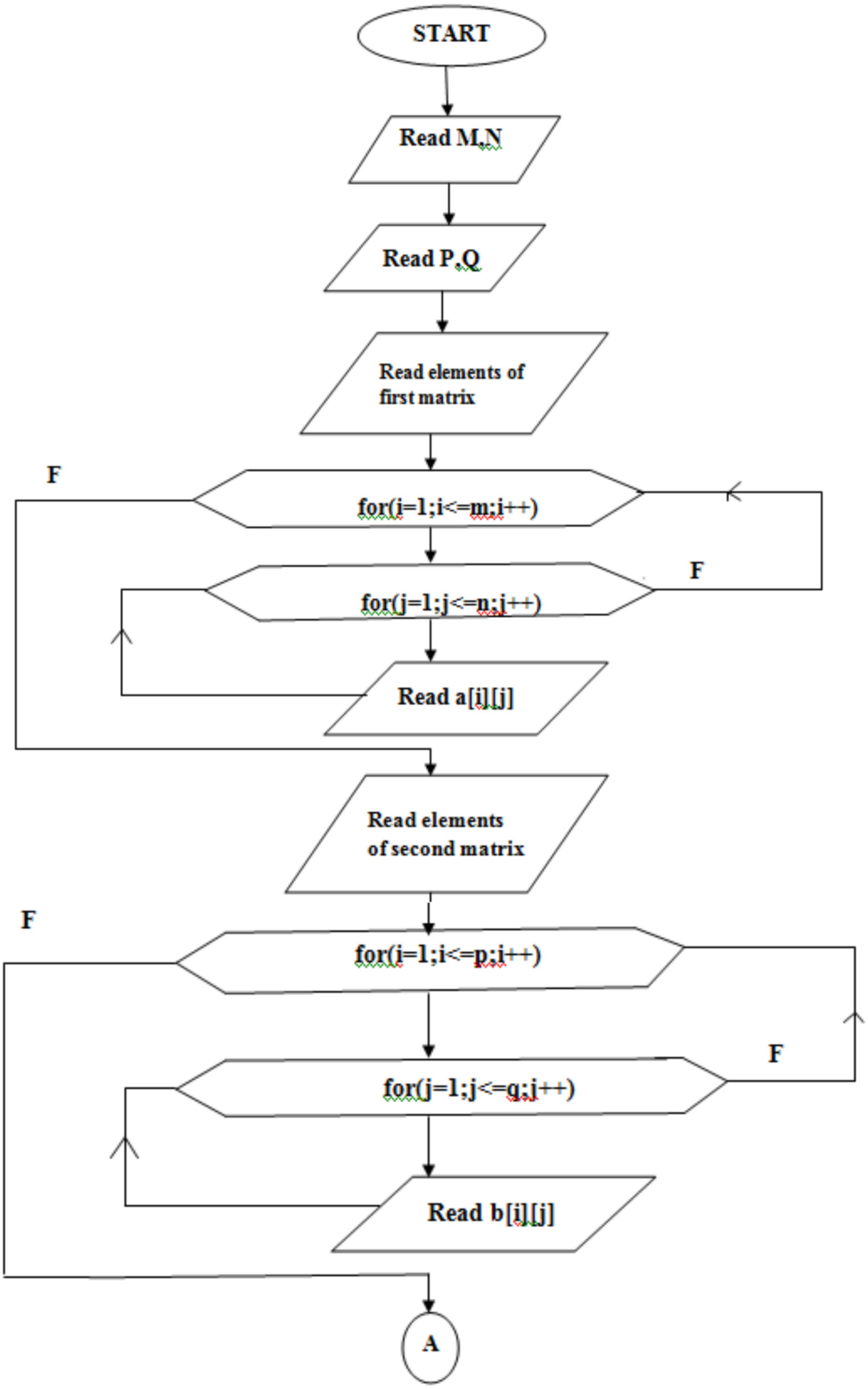
Step16: calculate c[i][j]=a[i][j]+b[i][j] and goto step 17

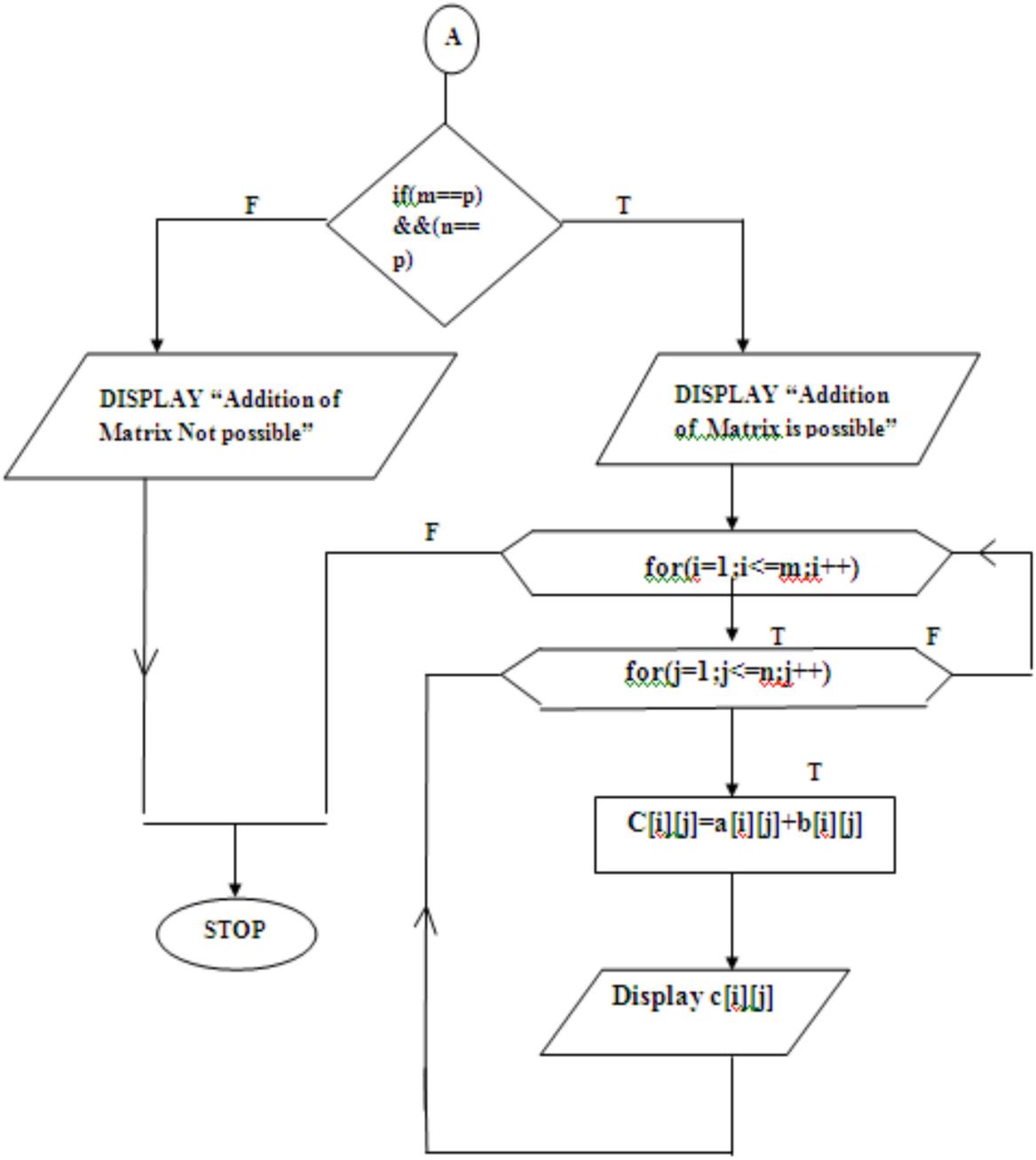
Step17: Display the value c[i][j] and goto step 15

Step18: Display matrix addition is not possible

Step19: Stop.

FLOWCHART




A
if(m==p)
&&(n==
p)
F
T
DISPLAY "Addition of
Matrix Not possible"
DISPLAY "Addition
of Matrix is possible"
F
for(i=1;i<=m;i++)
T
F
for(j=1;j<=n;j++)
T
C[i][j]=a[i][j]+b[i][j]
Display c[i][j]
STOP

PROGRAM:

```
#include <stdio.h>
// Function to take matrix input
void inputMatrix(int rows, int cols, int matrix[rows][cols]) {
    printf("Enter elements of the matrix (%dx%d):\n", rows, cols);
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("Element [%d][%d]: ", i + 1, j + 1);
            scanf("%d", &matrix[i][j]);
        }
    }
}

// Function to add two matrices
void addMatrices(int rows, int cols, int matrix1[rows][cols], int
matrix2[rows][cols], int result[rows][cols]) {
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            result[i][j] = matrix1[i][j] + matrix2[i][j];
        }
    }
}

// Function to display a matrix
void displayMatrix(int rows, int cols, int matrix[rows][cols]) {
    printf("The matrix is:\n");
```

```
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }
}

int main() {
    int rows, cols;

    // Input the dimensions of matrices
    printf("Enter number of rows and columns of the matrices: ");
    scanf("%d %d", &rows, &cols);

    int matrix1[rows][cols], matrix2[rows][cols], result[rows][cols];

    // Input matrices

    printf("\nInput first matrix:\n");
    inputMatrix(rows, cols, matrix1);
    printf("\nInput second matrix:\n");
    inputMatrix(rows, cols, matrix2);

    // Perform addition
    addMatrices(rows, cols, matrix1, matrix2, result);
```

```
    // Display result
    printf("\nResultant matrix after addition:\n");
    displayMatrix(rows, cols, result);

    return 0;
}
```

OUTPUT:

Enter number of rows and columns: 2 2

Input first matrix:

1 2

3 4

Input second matrix:

5 6

7 8

Resultant matrix

6 8

10 12

AIM: ii) To find the Multiplication of Two Matrices using c program

ALGORITHM

Step 1: START

Step2: Input values are a[10][10],b[10][10]

Step3: Read rows and columns of first matrix as m,n

Step4: Read rows and columns of second matrix as p,q

Step5: Initialize i=1,j=1,k=1

Step 6: Check the condition if(i<=m)then goto step7 Else goto step 9

Step7: Check the condition,if(j<=n)then goto step 8

Else goto step 6 after incrementing i

Step8: Read a[i][j] then increment j and goto step7

Step9: Check the condition ,if(i<=p)then goto step10 Else goto step12

Step10: Check the condition ,if(j<=q)then goto step11

Else increment i and goto step9

Step11: Read b[i][j] then increment j and goto step10

Step12: Check the condition if (n==p)then goto step13 Else goto step 22

Step13: Display matrix multiplication is possible then goto step14

Step14: Check the condition if (i<=m)then goto step15

Else goto step19

Step15: Check the condition if (j<=q)then goto step16

Else goto step14 after incrementing i

Step16: set c[i][j]=0 and goto step 17

Step17: Check the condition if (k<=n)then goto step18

Else increment j and goto step15

Step18: Calculate c[i][j]=c[i][j]+(a[i][k]*b[k][j]) and increment k and goto step 17

Step19: Check the condition ,if(i<=m) then check the condition if (j<=n) then goto

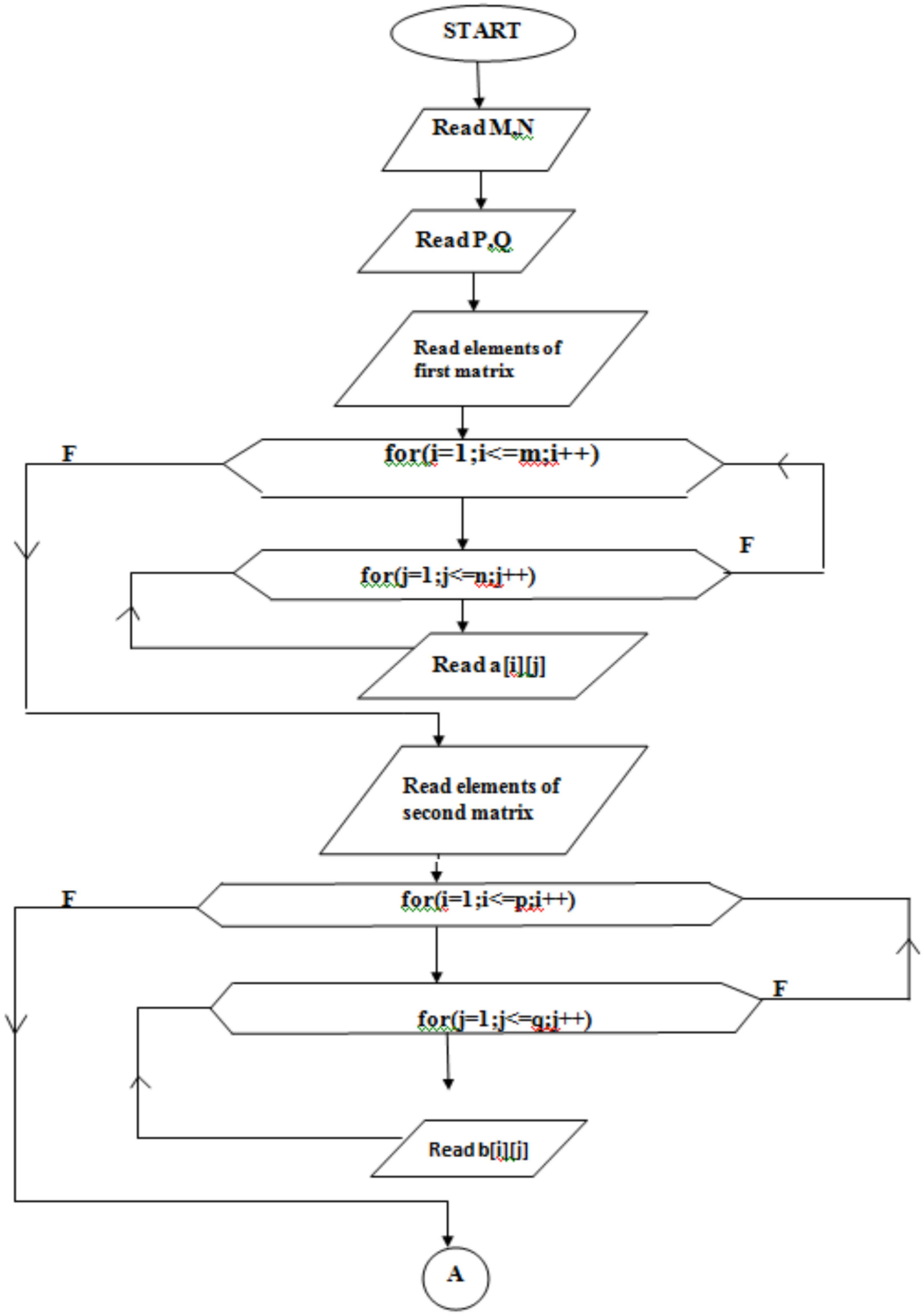
else increment i and repeat step 19

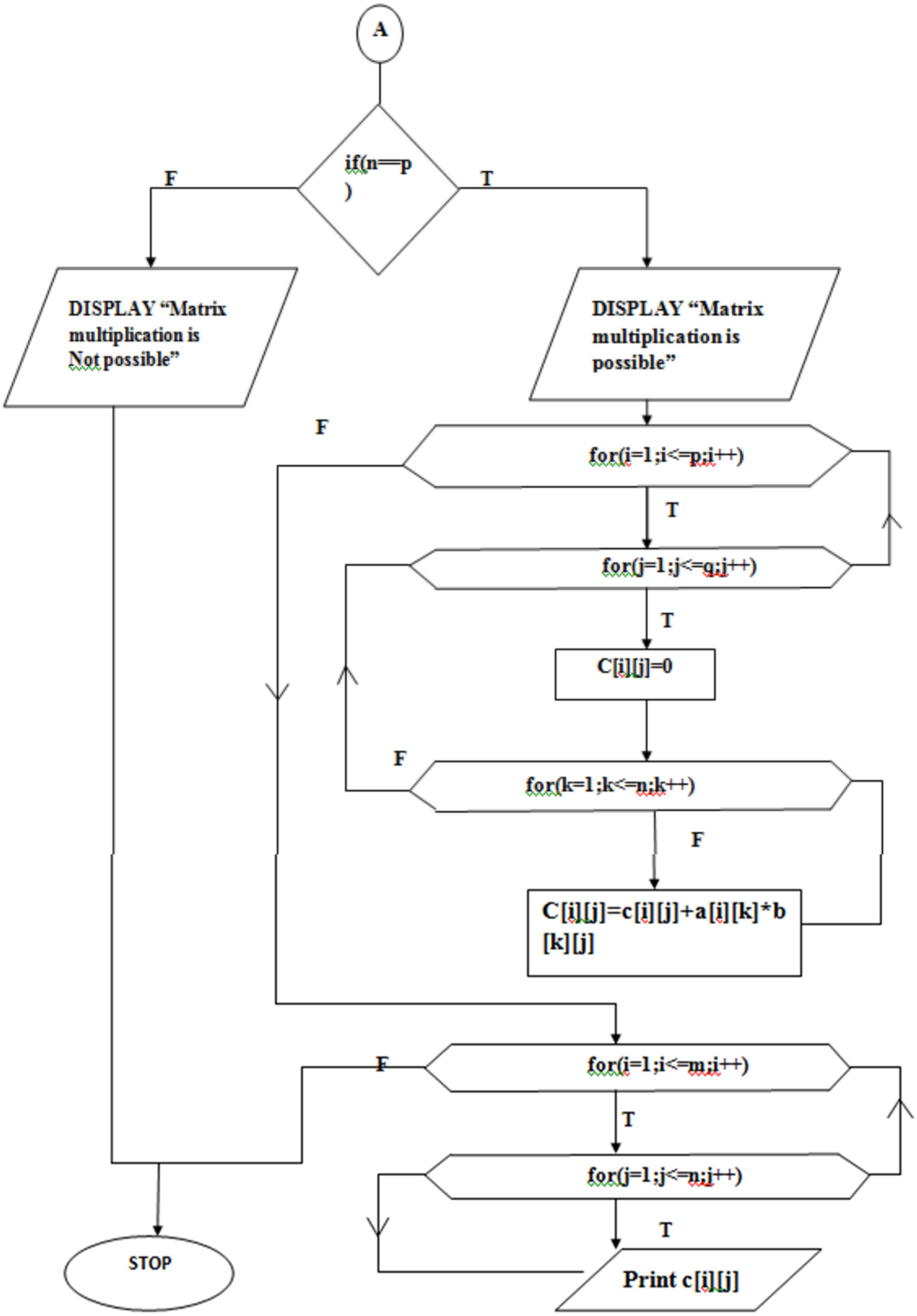
Step20: Display c[i][j] and increment j and goto step 19

Step21: Display matrix multiplication is not possible

Step22: Stop

FLOWCHART


START
Read M,N
Read P,Q
Read elements of first matrix
for(i=1;i<=m;i++)
F
for(j=1;j<=n;j++)
F
Read a[i][j]
Read elements of second matrix
for(i=1;i<=p;i++)
F
for(j=1;j<=q;j++)
F
Read b[i][j]
A


A
if(n==p
)
F
T
DISPLAY "Matrix multiplication is Not possible"
DISPLAY "Matrix multiplication is possible"
F
for(i=1;i<=p;i++)
T
for(j=1;j<=q;j++)
T
C[i][j]=0
F
for(k=1;k<=n;k++)
F
C[i][j]=c[i][j]+a[i][k]*b[k][j]
F
for(i=1;i<=m;i++)
T
for(j=1;j<=n;j++)
T
Print c[i][j]
STOP

PROGRAM:

```
#include <stdio.h>


// Function to take matrix input
void inputMatrix(int rows, int cols, int matrix[rows][cols]) {
   printf("Enter elements of the matrix (%dx%d):\n", rows, cols);
   for (int i = 0; i < rows; i++) {
      for (int j = 0; j < cols; j++) {
         printf("Element [%d][%d]: ", i + 1, j + 1);
         scanf("%d", &matrix[i][j]);
      }
   }
}


// Function to multiply two matrices
void multiplyMatrices(int r1, int c1, int r2, int c2, int matrix1[r1][c1], int
matrix2[r2][c2], int result[r1][c2]) {
   for (int i = 0; i < r1; i++) {
      for (int j = 0; j < c2; j++) {
         result[i][j] = 0;
         for (int k = 0; k < c1; k++) {
            result[i][j] += matrix1[i][k] * matrix2[k][j];
         }
      }
   }
```

```
}

// Function to display a matrix
void displayMatrix(int rows, int cols, int matrix[rows][cols]) {
    printf("The matrix is:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }
}

int main() {
    int r1, c1, r2, c2;

    // Input the dimensions of matrices
    printf("Enter rows and columns for first matrix: ");
    scanf("%d %d", &r1, &c1);
    printf("Enter rows and columns for second matrix: ");
    scanf("%d %d", &r2, &c2);

    // Check if multiplication is possible
    if (c1 != r2) {
        printf("Matrix multiplication not possible. Columns of first matrix must match
rows of second matrix.\n");
```

```
        return 1;
    }

    int matrix1[r1][c1], matrix2[r2][c2], result[r1][c2];

    // Input matrices
    printf("\nInput first matrix:\n");
    inputMatrix(r1, c1, matrix1);
    printf("\nInput second matrix:\n");
    inputMatrix(r2, c2, matrix2);

    // Perform multiplication
    multiplyMatrices(r1, c1, r2, c2, matrix1, matrix2, result);

    // Display result
    printf("\nResultant matrix after multiplication:\n");
    displayMatrix(r1, c2, result);

    return 0;
}
```

OUTPUT:

Enter number of rows and columns for first matrix: 2 3

Enter number of rows and columns for first matrix: 3 2

Input first matrix:

1 2 3

4 5 6

Input second matrix:

7 8

9 10

11 12

Resultant matrix

58 64

139 154

9. Write a C program for concatenation of two Strings

AIM: To Write a C program for concatenation of two Strings

ALGORITHM:

Step1: start

Step2: Declare variables a and b

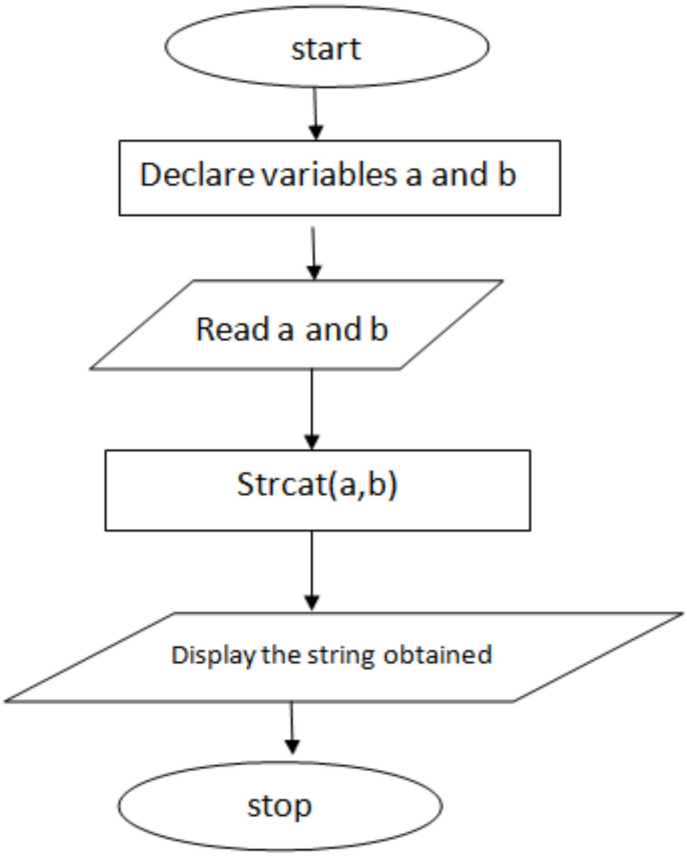
Step3: Read a and b

Step4: string operation Strcat(a,b)

Step5: Display the string obtained

Step6: stop

Flowchart:



PROGRAM:

```
#include <stdio.h>
#include <string.h>
int main( )
{
char a[100], b[100];
printf("Enter the first string\n");
gets(a);
printf("Enter the second string\n");
gets(b);
strcat(a,b);
printf("String obtained on concatenation is %s\n",a);
return 0;
}
```

Output:

```
Enter the first string
hi
Enter the second string
hello
String obtained on concatenation is hihello
```

10. Write a program for length of a string with and without string Handling functions

AIM: To Write a program for length of a string with and without string Handling functions

ALGORITHM:

Step1: Create a variable str, i, length.

Step2: Printf and scanf are used to accept the value.

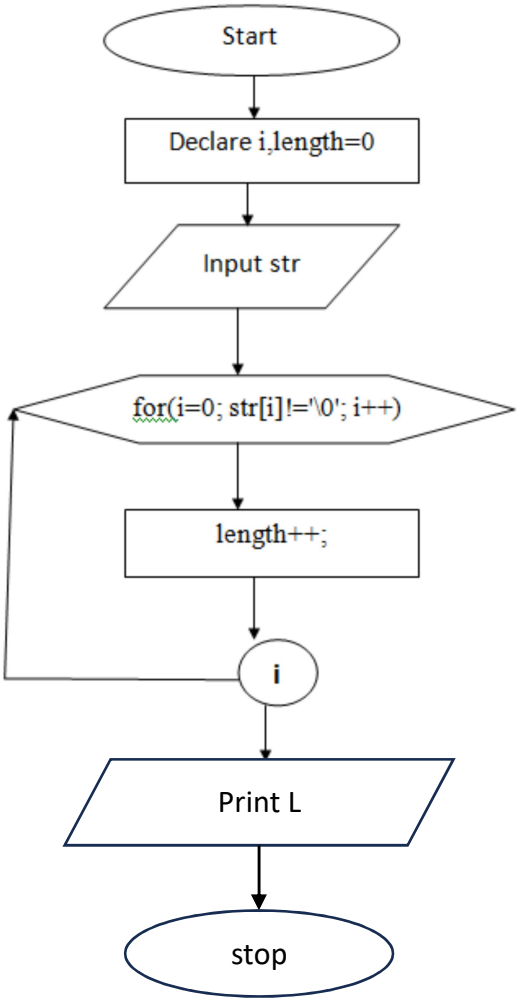
Step3: Start a for loop.

Step4: When null (‘0’), the loop should be terminated.

Step5: Print the length

Step 6: Stop

Flowchart:



PROGRAM:

```
#include<stdio.h>
int main()
{
char str[100];
int i,length=0;
 printf("Enter a string: \n");
 scanf("%s",str);
 for(i=0; str[i]!='\0'; i++)
 {
 length++;
 }
 printf("\nLength of input string: %d",length);
 return 0;
 }
```

OUTPUT:

```
Enter a string:
WELCOME
Length of input string: 7
```

11. Write a program to demonstrate Call by Value and Call by Reference mechanism

AIM: To Write a program to demonstrate Call by Value and Call by Reference mechanism

call by value

ALGORITHM:

step1: start

Step 2: Enter any 2 numbers at runtime

Step 3: Read those two numbers from console

Step 4: Call the function swap with arguments is a call by value

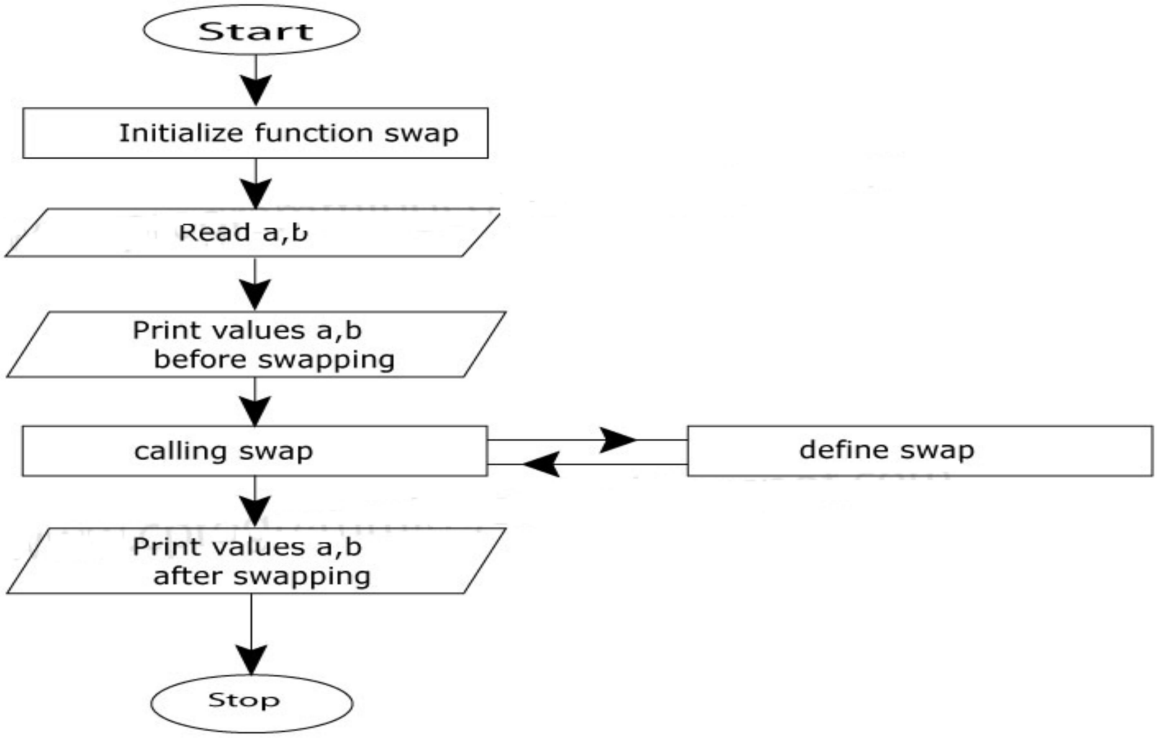
Step 5: Go to called function

swap(int a,int b)

Step 6: Print the numbers after swap

Step7: stop

FLOWCHART:



Program:

```
#include<stdio.h>
void main( )
{
void swap(int,int);
int a,b;
printf("enter 2 numbers");
scanf("%d%d",&a,&b);
printf("Before swapping a=%d b=%d",a,b);
swap(a,b);
printf("after swapping a=%d, b=%d",a,b);



}
```

```
void swap(int a,int b)
{
int t;
t=a;
a=b;
b=t;
}
```

Output

enter 2 numbers 10 20

Before swapping a=10 b=20

After swapping a=10 b=20

call by reference

ALGORITHM:

step1: start

Step 2: Enter any 2 numbers at runtime

Step 3: Read those two numbers from console

Step 4: Call the function swap with arguments is a call by reference

Step 5: Go to called function

swap(int a,int b)

Step 6: Print the numbers after swap

Step7: stop

PROGRAM:

```
#include<stdio.h>
void main( )
{
void swap(int *,int *);
int a,b;
printf("enter 2 numbers");
scanf("%d%d",&a,&b);
printf("Before swapping a=%d b=%d",a,b);
swap(&a, &b); //address are sent as an argument
printf("after swapping a=%d, b=%d",a,b);
}
void swap(int *a,int *b)
{
int t;
t=*a;
*a=*b;
*b=t;
}
```

Output:

enter 2 numbers 10 20

Before swapping a=10 b=20

After swapping a=20 b=10

12. Write a program to find GCD of Two numbers using Recursion

AIM: To write a program to find GCD of Two numbers using Recursion.

ALGORITHM:

Step 1 − Define the recursive function.

Step 2 − Read the two integers a and b.

Step 3 − Call recursive function.

if i>j

then return the function with parameters i,j

if i==0

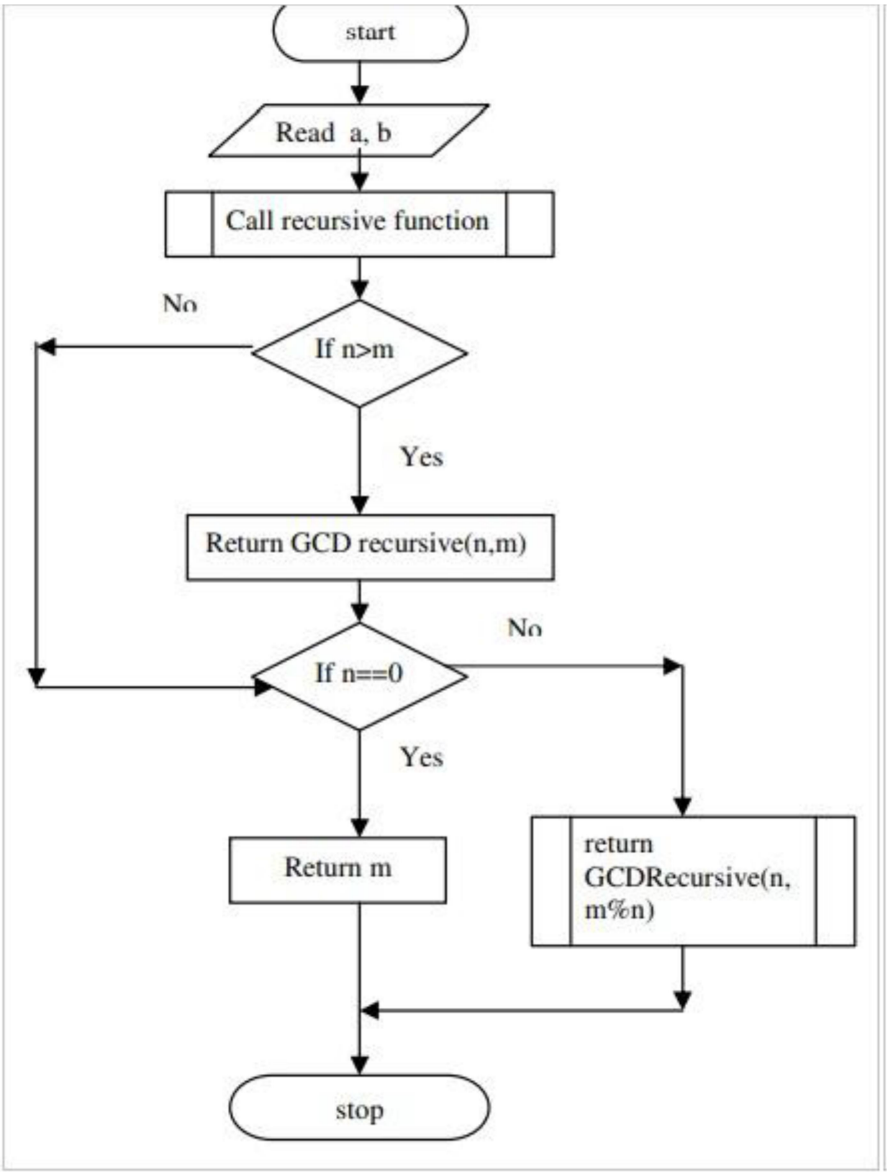
then return j

else

return the function with parameters i,j%i.

Step 4: Stop.

FLOWCHART:


start
Read a, b
Call recursive function
No
If n>m
Yes
Return GCD recursive(n,m)
No
If n==0
Yes
Return m
return GCDRecursive(n, m%n)
stop

Program:

```
#include<stdio.h>
#include<math.h>
unsigned int GCD(unsigned i, unsigned j);
int main( )
{
int a,b;
printf("Enter the two integers:");
scanf("%d%d",&a,&b);
printf("GCD of %d and %d is %d",a,b,GCD(a,b));
return 0;
}
/* Recursive Function*/
unsigned int GCD(unsigned i, unsigned j)
{
if(j>i)
  return GCD(j,i);
  if(j==0)
  return i;
  else
  return GCD(j,i%j);
  }
```

Output

```
Enter the two integers: 4 8
GCD of 4 and 8 is 4
```

13. Write a c program to perform various operations using pointers.

AIM: To Write a c program to perform various operations using pointers.

Algorithm:

Step 1: start

Step 2: Declare integer variables

Step 3: Variables for storing arithmetic operations solution

Step 4: Pointer variables for variables a and b

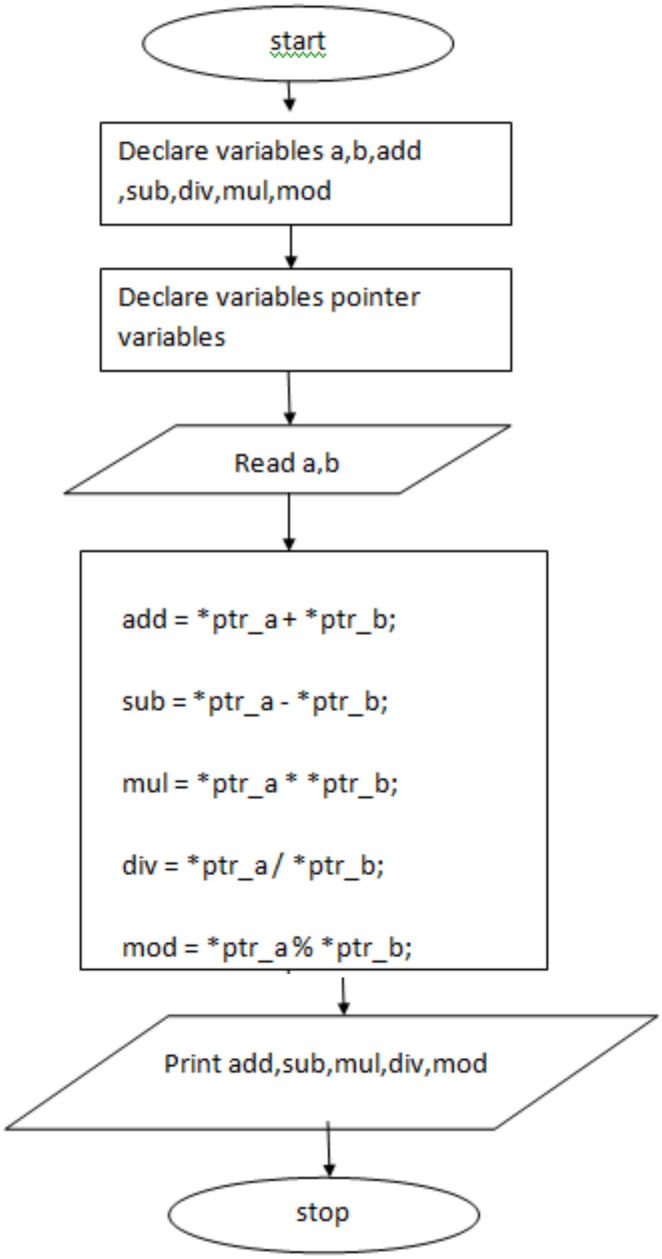
Step 5: Initialization of pointers

Step 6: Performing arithmetic Operations on pointers

Step 7: printing values

Step 8: stop

FLOWCHART


start
Declare variables a,b,add ,sub,div,mul,mod
Declare variables pointer variables
Read a,b
add = *ptr_a + *ptr_b;
sub = *ptr_a - *ptr_b;
mul = *ptr_a * *ptr_b;
div = *ptr_a / *ptr_b;
mod = *ptr_a % *ptr_b;
Print add,sub,mul,div,mod
stop

Arithmetic Operations

```
#include <stdio.h>
int main( )
{
int a = 20, b = 10;
int add, sub, div, mul, mod;

int *ptr_a, *ptr_b;
ptr_a = &a;
ptr_b = &b;
add = *ptr_a + *ptr_b;
sub = *ptr_a - *ptr_b;
mul = *ptr_a * *ptr_b;
div = *ptr_a / *ptr_b;
mod = *ptr_a % *ptr_b;
// Printing values
printf("Addition = %d\n", add);
printf("Subtraction = %d\n", sub);
printf("Multiplication = %d\n", mul);
printf("Division = %d\n", div);
printf("Modulo = %d\n", mod);
return 0;
}
```

Output:

Addition = 30

Subtraction = 10

Multiplication = 200

Division = 2

Modulo = 0

14. Write a c program to read data of 10 employees with a structure of 1.employee id2.aadar no, 3.title, 4.joined date, 5.salary, 6.date of birth, 7.gender, 8.department

AIM: To write a c program to read data of 10 employees with a structure of 1.employee id2.aadar no, 3.title, 4.joined date, 5.salary, 6.date of birth, 7.gender, 8.department

Algorithm: Read and Display Employee Data

Step 1: Define the Structure

1. Create a structure named Employee with the following fields:
    - Employee ID (emp_id)
    - Aadhaar Number (aadhaar_no)
    - Title (title)
    - Joined Date (joined_date)
    - Salary (salary)
    - Date of Birth (dob)
    - Gender (gender)
    - Department (department)

Step 2: Declare Variables

2. Define an array of 10 Employee structures.

Step 3: Read Employee Data

3. Use a loop (for loop) to iterate through all employees.
4. For each employee, prompt the user to enter the required details.
5. Store the input data in the respective structure fields.

Step 4: Display Employee Data

6. Use another loop to iterate through all employees.
7. Print each employee's details in a readable format.

Step 5: End

8. Exit the program.

PROGRAM:

```
#include <stdio.h>



#define EMP_COUNT 10



// Define structure for Employee

typedef struct {

    int emp_id;

    long long aadhaar_no;
```

```
    char title[50];
    char joined_date[15];
    float salary;
    char dob[15];
    char gender;
    char department[50];
} Employee;


void inputEmployeeData(Employee emp[], int count) {
    for (int i = 0; i < count; i++) {
        printf("\nEnter details for Employee %d:\n", i + 1);
        printf("Employee ID: ");
        scanf("%d", &emp[i].emp_id);
        printf("Aadhaar Number: ");
        scanf("%lld", &emp[i].aadhaar_no);
        printf("Title: ");
```

```
        scanf(" %[^"]s", emp[i].title);
        printf("Joined Date (DD/MM/YYYY): ");
        scanf(" %s", emp[i].joined_date);
        printf("Salary: ");
        scanf("%f", &emp[i].salary);
        printf("Date of Birth (DD/MM/YYYY): ");
        scanf(" %s", emp[i].dob);
        printf("Gender (M/F): ");
        scanf(" %c", &emp[i].gender);
        printf("Department: ");
        scanf(" %[^"]s", emp[i].department);
    }
}

void displayEmployeeData(Employee emp[], int count) {
    printf("\nEmployee Details:\n");
```

```
    for (int i = 0; i < count; i++) {
        printf("\nEmployee %d:\n", i + 1);
        printf("ID: %d\n", emp[i].emp_id);
        printf("Aadhaar No: %lld\n", emp[i].aadhaar_no);
        printf("Title: %s\n", emp[i].title);
        printf("Joined Date: %s\n", emp[i].joined_date);
        printf("Salary: %.2f\n", emp[i].salary);
        printf("Date of Birth: %s\n", emp[i].dob);
        printf("Gender: %c\n", emp[i].gender);
        printf("Department: %s\n", emp[i].department);
    }
}

int main() {
    Employee employees[EMP_COUNT];
```

```
    // Input employee data
    inputEmployeeData(employees, EMP_COUNT);


    // Display employee data
    displayEmployeeData(employees, EMP_COUNT);


    return 0;
}
```

OUTPUT:

15. Write a Program to demonstrate dynamic arrays using Dynamic Memory Management functions

AIM: To Write a Program to demonstrate dynamic arrays using Dynamic Memory Management functions

ALGORITHM:

Step 1: start

Step 2: declare variables

Step 3: read size of elements

Step 4: Memory allocates dynamically using malloc( )

Step 5: Checking for memory allocation

if (ptr == NULL)

Memory not allocated

else

Memory allocated

Step 6: using for loop Get the elements of the array

Step 7: Print the elements of the array

Step 8: stop.

Program:

```
#include <stdio.h>
#include <stdlib.h>
int main( )
{
// address of the block created hold by this pointer
int* ptr;
int size;
printf("Enter size of elements:");
scanf("%d", &size);
ptr = (int*)malloc(size * sizeof(int));
if (ptr == NULL)
{
printf("Memory not allocated.\n");
}
else
{
printf("Memory successfully allocated using " "malloc.\n");
for (int j = 0; j < size; ++j)
{
ptr[j] = j + 1;
}


printf("The elements of the array are: ");
for (int k = 0; k < size; ++k)
```

```
{
printf("%d, ", ptr[k]);
}
}
return 0;
}
```

Output:

Enter size of elements:5

Memory successfully allocated using malloc.

The elements of the array are: 1, 2, 3, 4, 5,