Practical 1:

Aim: Write a C program to understand basic datatypes & I/O.

Theory:

1] Write a program to display student's name, roll no, mobile no & percentage.

Algorithm:

Step 1: Declare a variable name roll no as integer, also declare name, mobile no as character & percentage as float.

Step 2: Use printf function to print questions for user in order to give input.

Step 3: Use scanf function to read user's input and store in its allocated memory.

Step 4: Again use printf function to display the output.

• CODE:

```
// basic datatypes
# include <stdio.h>
# include <conio.h>
void main()
{
        int roll_no;
        char name[20], mobile_no[10];
        float percentage;
        clrscr();
        printf("Enter student's name:\n");
        scanf("%s", &name);
        printf("Enter student's roll no:\n");
        scanf("%d", &roll_no);
        printf("Enter student's mobile no:
                \n");
        ~~printf("~~
        scanf("%s", & mobile_no);
        printf("Enter student's percent-
                age:\n");
        ~~printf~~
        scanf("%f", &percentage);
        printf("Student's name: \n"
                %s", name);
        printf("Student's roll no: %d",
                roll_no);
```

<u>Conclusion</u>: The given program gives us an idea about how built in datatypes work in C and also about how user can give input & display output.

[illegible]

```
printf ("student's mobile no: %.2 \n",
                      mobile_no);
printf ("student's percentage: %f\n",
                   percentage);
getch();
}
```

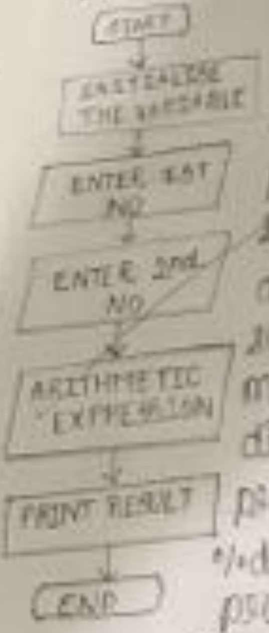
Output:

```
Enter student's name:
Ankita
Enter student's rollno:
1867
Enter student's percentage:
89.12
Enter student's mobile no:
9999999999

student's Name: Ankita
student's roll no: 1867
student's mobile no: 9999999999
student's percentage: 89.120003
```

ii - CODE:

```
//dynamic calculator
# include <stdio.h>
# include <conio.h>
void main()
{
        int num1, num2;
        float add, sub, mult, div;
        clrscr();
        printf ("Enter first number :\n");
        scanf ("%d", &num1);
        printf ("Enter second number :\n");
        scanf ("%d", &num2);
        add = num1 + num2;
        sub = num1 - num2;
        mult = num1 * num2;
        div = num1/num2;
        printf (" Addition of %d and
        %d is %f\n", num1, num2, add);
        printf ("Subtraction of %d and
        %d is %f\n", num1, num2, sub);
        printf ("Multiplication of %d
        and %d is %f\n", num1, num2, mult);
        printf ("Division of %d and %d
        is %f\n", num1, num2, div);
        getch();
}
```

FLOWCHART:



Practical 2:

Aim: Write a C program on operator and expression.

Theory:

i) Write a program to create a dynamic calculator.

Algorithm:

Step1: Declare a variable name for first & second number of integer.

Step2: Now use scanf function to receive input from user.

Step3: Now to add two numbers given by user, use the expression num1+num2

Step4: Now to subtract two numbers given by user, use expression num1 - num2

Step5: Again use expression num1*num2 if user wishes to multiply the two inputs.

Step6: use expression num1/num2 if user wishes to divide the two inputs.

Step7: Now use printf function to display output.

6] write a program in c to expl-
ain ternary operator.

Algorithm:

Step1: Declare variables a, b & x
as integers

Step2: store the value of a as
5 & store the variable
value of b as 15

Step3: Now to bet compare between
who is greater use
ternary operator x to
find

Step4: use printf function to
display output

conclusion: these programs help
us ~~to~~ in having better
understanding about operators
and expressions.

[illegible]

• output:

```
Enter first number: 3
Enter second number: 3
Addition of 3 and 3 is 6.0000
subtraction of 3 and 3 is 0
Multiplication of 3 and 3 is 9.0000
Division of 3 and 3 is 1.0000
```

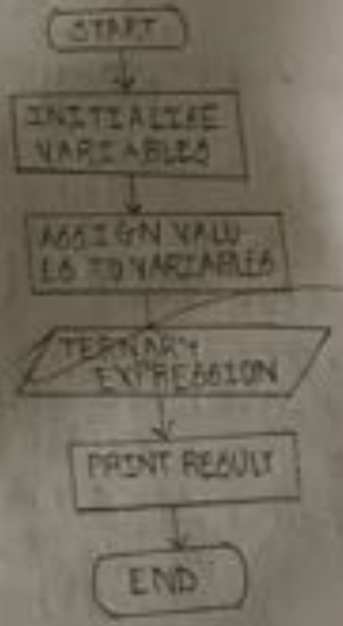
6] CODE:

```
// ternary operator
# include<stdio.h>
# include<conio.h>
void main()
{
    int a,b,x;
    clrscr();
    a = 5;
    b = 15;
    x = (a>b) ? a:b;
    printf (x);
    getch();
}
```

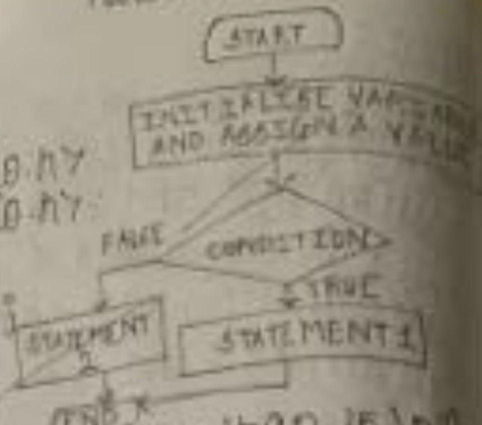
• output

```
15
```

FLOWCHART:



FLOWCHART:



a] · CODE:
```
// if statement
#include<stdio.h>
#include<conio.h>
void main()
{
        int i=10;
        clrscr();
        if (i>15)
        { printf("10 is less than 15 \n");
        }
        printf("I am not in if \n");
        getch();
}
```
· output:
```
I am not in if.
```
b] · CODE:
```
// if else statement
#include<stdio.h>
#include<conio.h>
{
        int i=20;
        clrscr();
        if (i<15)
        { print ("20 is smaller than
                15\n");
        }
```

Practical 3:

Aim: Write a program in c on decision statement (if, if else, nested if)

Theory:

a] Write a program in c to explain if statement

Algorithm:

Step 1: Declare a variable as integer and assign its value i.e. 20

Step 2: Now to compare whether 20 is greater than 15 use if statement

Step 3: If the condition is true, print that 20 is less than 15 & if condition is false, skip the if statement & print ~~so~~ I am not in if

b] Write a program in c to explain if else statement

Step 1: Declare a variable as integer and assign its value i.e. 20

Step 2: Now to compare the given value if its greater or not use if else conditional statement.

Step 3: If condition is true then print 20 is less than 15 or if condition is false then print 20 is greater than 15

Q] Write a program in c to explain nested if statement

Algorithm:

Step 1: Declare a variable as integer and assign value ie 20

Step 2: Now use nested if logic to compare if given no is greater or not

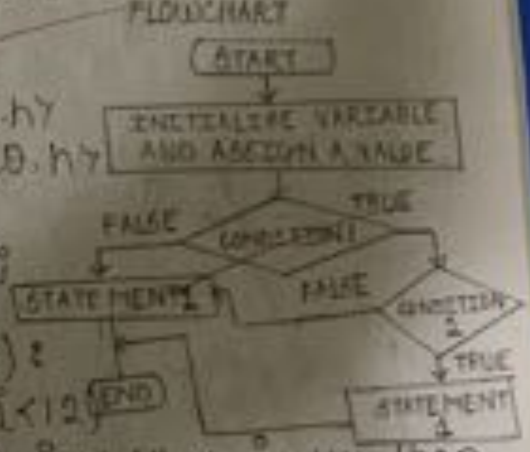
Step 3: If first condition is true then go to second condition if second condition is also true then print that 20 is greater than 15 & 12

```
else{
{ printf("20 is greater than 15\n");
}getch();
}
```

output:
20 is greater than 15

CODE:
```
// nested if
# include<stdio.h>
# include<conio.h>
void main
{
      int i= 20;
      clrscr();
      if (i<15){
      {   if (i<12)
          {  printf("20 is less than
                15 & 12\n");
          }
      }
      else{
      { printf("20 is greater than
                15 & 12\n");
      }
getch();
}
```

FLOWCHART



· output:
20 is greater than 10 & 12.

one of
if conditions are not true
then skip the part & print
20 is greater than 15 & 12.

conclusion: These programs help us to
understand the working
of if, if else & nested if
conditional statements.

[illegible]

Practical 4

Aim: To display prime number using for loop.

Algorithm:

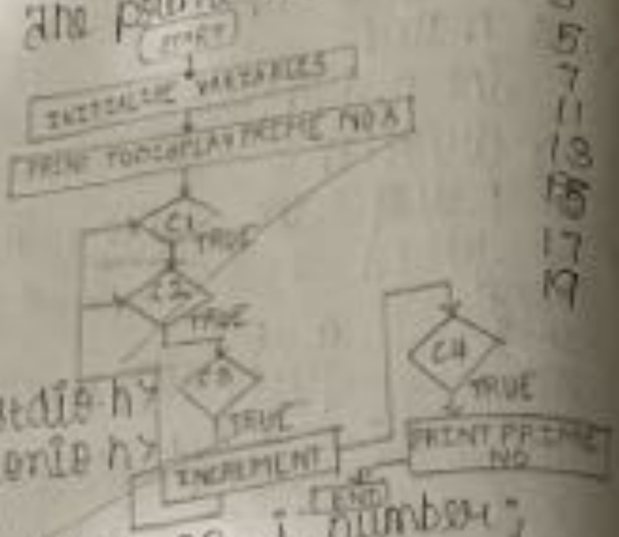
```
step1: Initiage three variables out
       of which two are loop
       variable and one is a
       count variable

step2: Initialize a for loop from
       from 2 to 50 let the
       count variable to zero

step3: Put another loop within the
       loop in step 2 that goes
       from 2 to the first loop
       variable % 2

step4: Use the if conditional
       statement to check whether
       (1st loop variable % 2nd
       variable ==0) if true,
       Increment count variable
       by 1

step5: Come out of the second loop
       and check whether the count
       variable is 0 if true print
       the number
```

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{    int n,i,a;
     clrscr();
     printf(" The prime numbers are : ");
     for (i=2; i<=20; i++)
     {      a=0;
            for(n=2; n<(i+1)/2; n++);
        {         if (i%n==0)
                 { a++;
                 }
        }
         if(a==0)
        { printf("%d \n",i);
        }
     }
     getch();
}
```

Output : The prime numbers are : 2
3
5
7
11
13
15
17
19

Flowchart:



Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{ int n1=0, n2=1, n3, i, number;
  clrscr();
  printf("Enter number of elements:");
  scanf("%d", &number);
  for printf("%d", n1);
  for (i=2; i<=number; i++)
{    n3=n1+n2;
     printf("\n%d", n3);
     n1=n2;
     n2=n3;
}
getch();
```

Step 6: Terminate the program.

Conclusion: Prime numbers were displayed using for loop.

Aim: Create a c program on fibonacci series.

Algorithm:

Step 1: Start the turbo c

Step 2: Declare variable n1, n2, n3, i, number

Step 3: Initialize variable n1=0, n2=1, number=0

Step 4: Enter the number of terms of fibonacci series to be printed

Step 5: Print first two term of series as n1=0 & n2=1

Step 6: Use the for loop as per following step
$n3 = n1 + n2;$
$n1 = n2;$
$n2 = n3;$
Increase the value of i element each time by 1.

Step 7: Print the value of number

step 8: End the program

Conclusion: Thus we have successfully executed fiboracci series in turbo c.

c] Aim: Write a c program on following expression

```
1
2  3
4  5  6
7  8  9  10
11 12 13 14 15
```

Algorithm:

Step 1: Start the turbo C program

step 2: Declare the variable rows, i, j, number = 0;

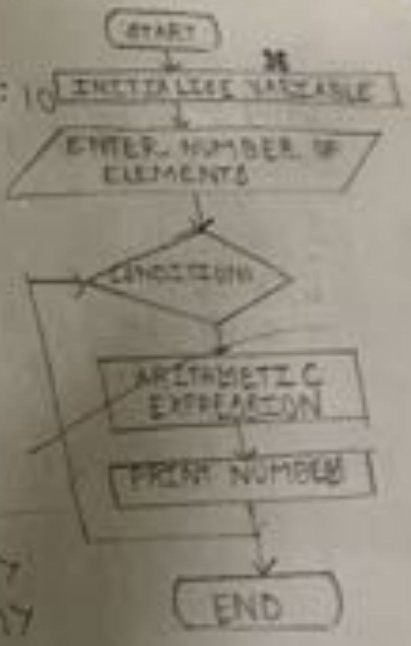
step 3: Display the number of rows

step 4: Enter for loop as i = 1; i<= rows ; i++

Output:

```
Enter no of elements : 10
0
1
1
2
3
5
8
13
21
34
55
```

FLOWCHART:



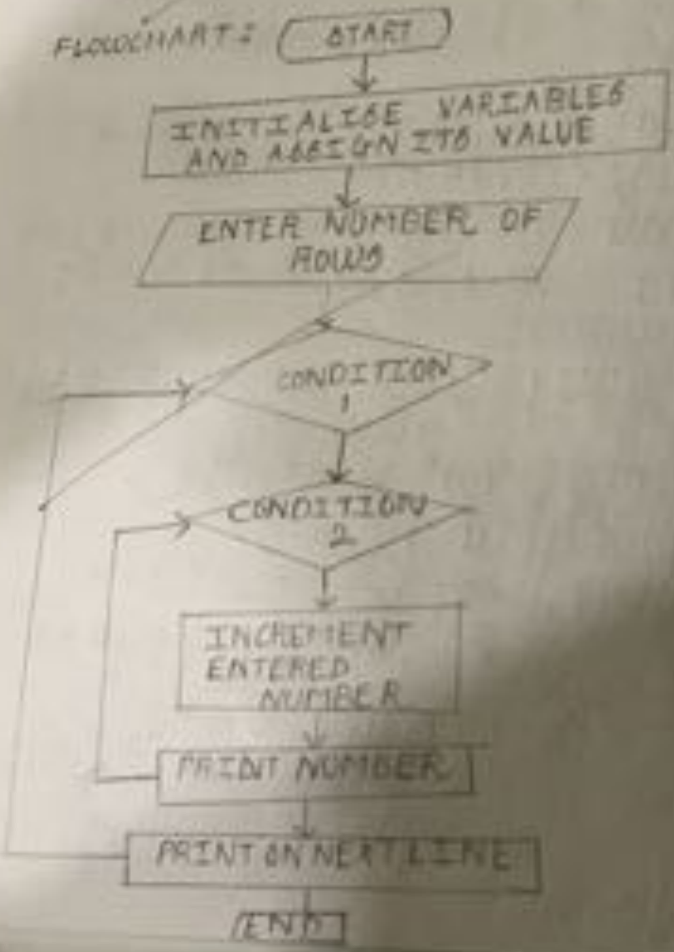
Program:

```
# include < stdio.h>
# include < conio.h>
void main()
{
    int rows, i, j, number = 0;
    clrscr();
    printf ("Enter the number of
                rows:");
    scanf ("%d", & rows);
    printf("\n");
    for ( i = 0; i<= rows ; i++)
    {   for ( j = 0; j<= i ; j++)
        {
            number ++ ;
            printf ("%d ", n));
        }
        printf ("\n");
    }
    getch();
}
```

output :

```
Enter number of rows : 4
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

FLOWCHART :



step 5: Create nested for loop as j=1; j<=i; j++;

step 6: Display the number as per user entries the sequence from i=1

step 7: Increment the number from 1

step 8: Display space

step 9: End the program.

Conclusion : Thus we have successfully executed given expression in turbo C using nested for loop.

[illegible]

Practical 5:

Aim: write a program to print a number that is odd or even number in array

Algorithm

Step 1: Create an array. Take its size from user & define its element using loop

Step 2: Display size of array from

Step 3: Display element of array entered by user

Step 4: Take the initiator in for loop in which all elements in array exits

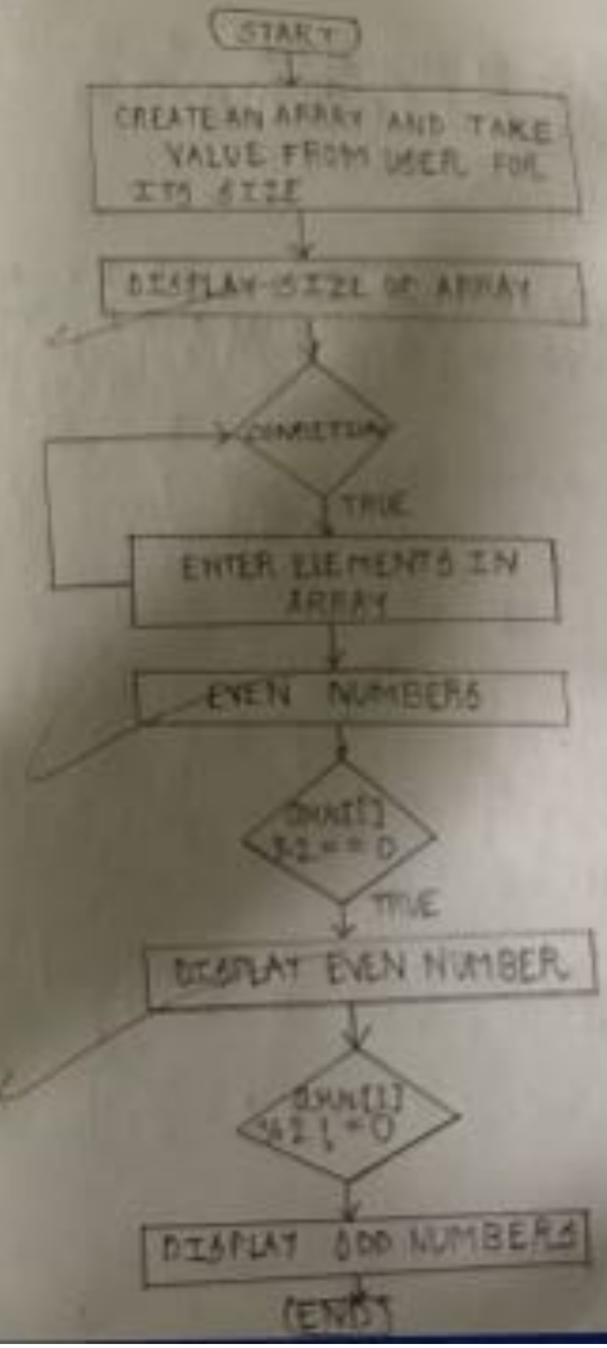
Step 5: Display even number from array using for loop if (arr[i] % 2 == 0) then

Step 6: display the even number in given array

Step 6: Display odd number from array using ~~[illegible]~~ for loop if (arr[i] % 2 != 0 then display odd numbers in given array

Step 7: Terminate the program

FLOWCHART:



```
OUTPUT:
Enter size of array : 5
Enter elements in array : 22
44
55
66
77
Even numbers in array : 22  44  66
Odd numbers in array : 55  77
```

Code:

```
#include<stdio.h>
#include<conio.h>
void main()
{
     int array[100], i, num;
     clrscr();
     printf("enter the size of array:");
     scanf("%d",&num);
     printf(" enter the elements of
       array \n");
   for (i=0; i<num; i++)
   {  scanf("%d", & array[i]);
   }
   printf(" Even numbers in array");
   for(i=0; i<num; i++)
   {    if (array[i]%2==0)
        {    printf(" %d \t", array[i]);
        }
   }
   printf(" Odd numbers in array ");
   for (i=0; i<num; i++)
   {    if (array[i] % 2 !=0)
        {    printf(" %d \t", array[i]);
        }
   }
   getch();
}
```

Conclusion : successfully executed the program

Aim: Write a C program to find largest number among array

Algorithm:

Step 1: Start turbo C.

Step 2: Declare the variable i and integer in array a[10]

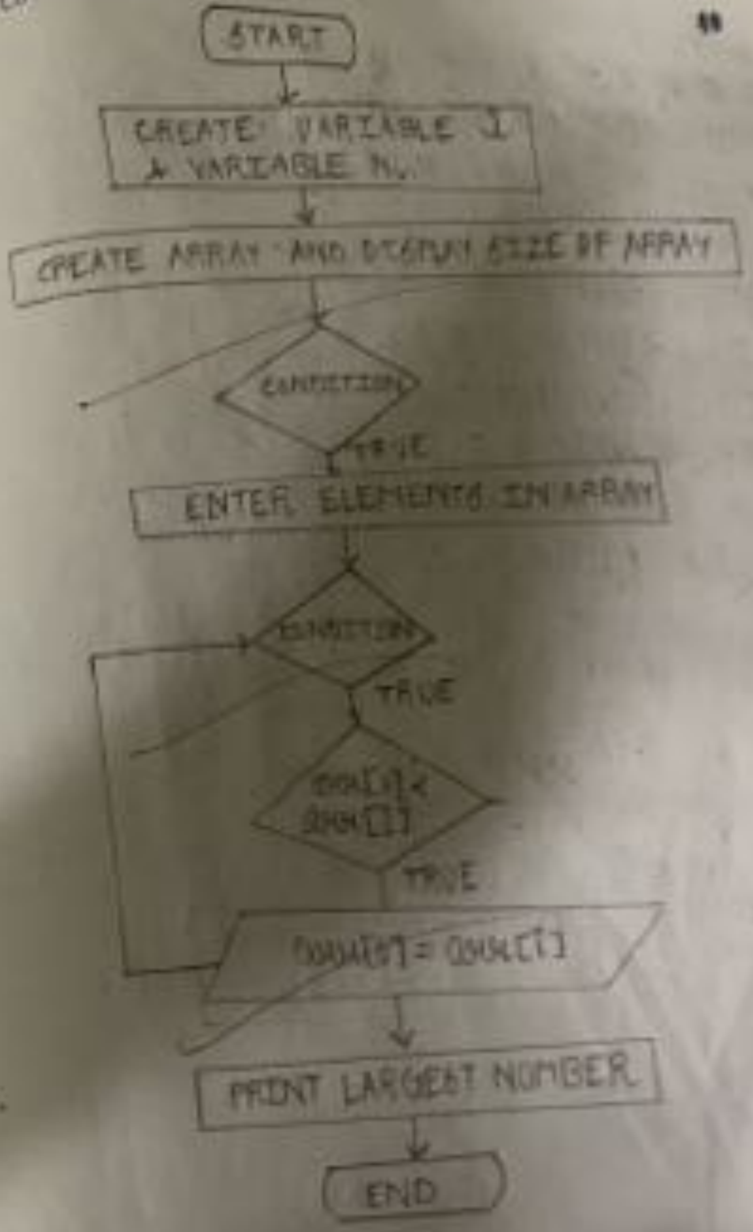
Step 3: Enter the for loop at i=0; i<10 & use the value of a[i] till i<10 exit the for loop

Step 4: Enter the for loop at i=0; i<10 use if conditional statment to check if a[0] <a[i] if true, put a[0] = a[i]

Step 5: Run the above for loop for i<10, exit the loop

Step 6: terminate the program

FLOW CHART:



OUTPUT:
~~entering size of array: 5~~
~~enter elements in array:~~

```
Enter the number of elements
  from (1 to 100):  5
Enter elements : 21
Enter elements : 22
Enter elements : 32
Enter elements : 20
Enter elements : 17
Largest number is 32.0000
```

code:

```
#include<stdio.h>
#include<conio.h>
void main ()
{
        int i, n;
        float arr[100];
        clrscr();
        printf("enter the number of
        element from (1 to 100): ");
        scanf("%d",&n);
        for(i = 0; i<n; i++)
        {
            printf("\n Enter elements:"
                         i+1);
            scanf("%f", & arr[i]);
        }
        for (i=1; i<n; i++)
        {
            if ( arr[0] < arr[i])
                 arr[0] = arr[i];
        }
        printf("\n largest number is %.2
                        arr[0]);
        getch();
}
```

Conclusion : successfully executed the program

Aim: write a C program to find sum and average of elements in array

Algorithm:

step1: Create an array, take its size from user & define its element in for loop

step 2: Now check if size is (n>100 || n<1) then print error

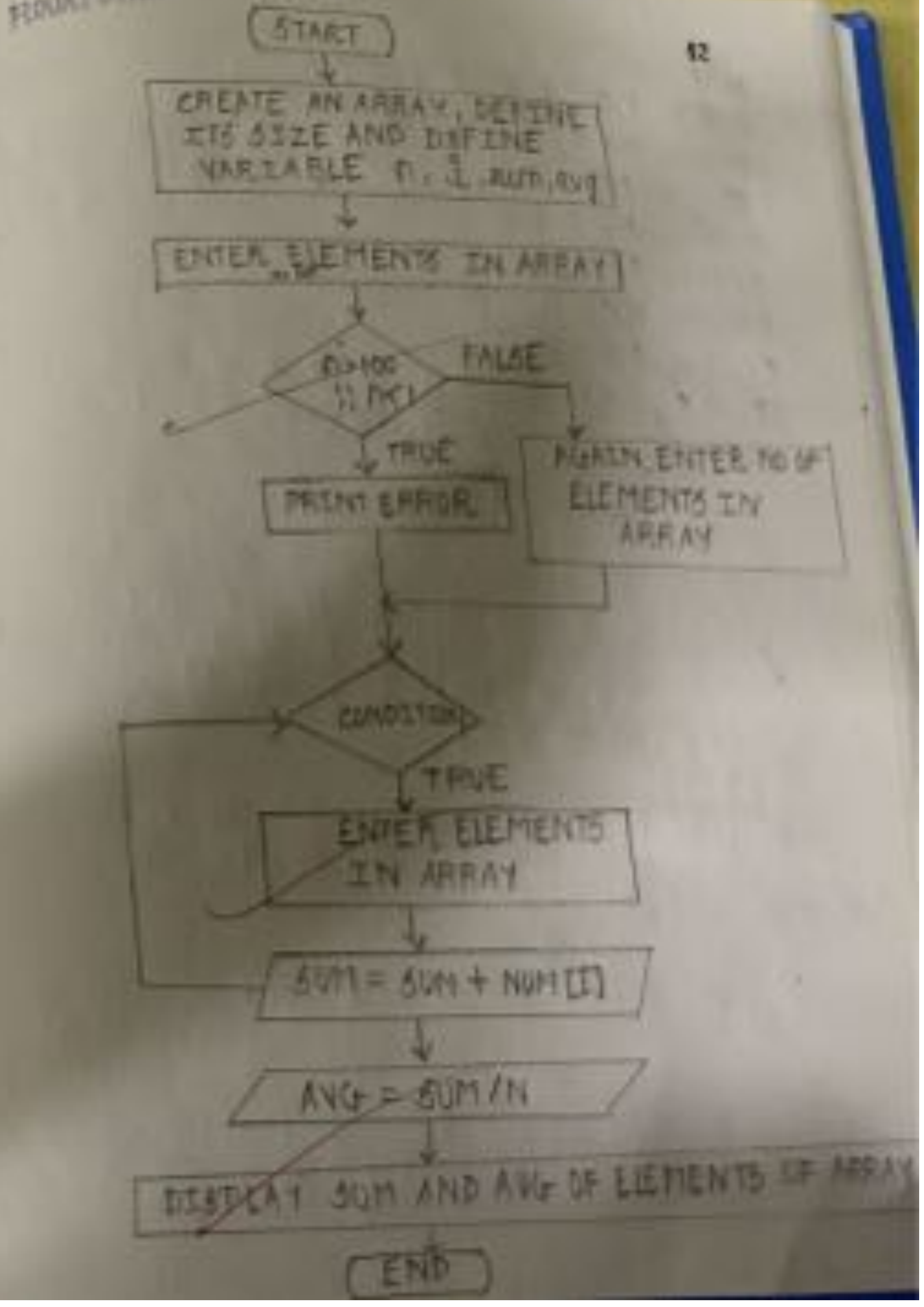
step 3: Now enter elements in array using for loop & find the sum ie sum = sum + num[i]

step 4: Find average by avg = sum/n

step 5: Display average & sum of elements in array

step 6: Terminate the program

FLOWCHART:


START
CREATE AN ARRAY, DEFINE ITS SIZE AND DEFINE VARIABLE n, i, sum, avg
ENTER ELEMENTS IN ARRAY
n>100 || n<1
FALSE
TRUE
PRINT ERROR
AGAIN ENTER NO OF ELEMENTS IN ARRAY
CONDITION
TRUE
ENTER ELEMENTS IN ARRAY
SUM = SUM + NUM[I]
AVG = SUM/N
DISPLAY SUM AND AVG OF ELEMENTS OF ARRAY
END

28

OUTPUT:
```
enter no of elements : 5
enter no of elements : 5
enter no : 1
enter no : 2
enter no : 3
enter no : 4
enter no : 5.
average of array is 3.0000
sum of array is 15.0000
```

code:
```
#include<stdio.h>
#include<conio.h>
void main ()
{
        int n, i;
        float num[100], sum, avg;
        clrscr();
        printf ("enter the no of elements");
        scanf ("%d", &n);
        while (n>100 || n<1)
      {   printf ("Error!");
      }
     printf (" enter number again");
     scanf ("%d", &n);
    for (i=0; i<n; i++)
    {
        printf (" Enter no:", i+1);
        scanf (" %f", &num[i]);
        sum= sum + num[i];
    }
    avg = sum/n;
   printf (" average of array is %f",
                avg);
   printf (" sum of array is %f", sum);
   getch();
}
```
Conclusion: successfully executed the program.

Practical 6

Aim: Programs on functions

Q.II write a C program to find factorial of a number using recursion

Algorithm:

Step 1: Define a function called factorial with argument n as integer

Step 2: In this function use if conditional statement to check whether the number is greater than 1 then return n * factorial (n-1) else return 1

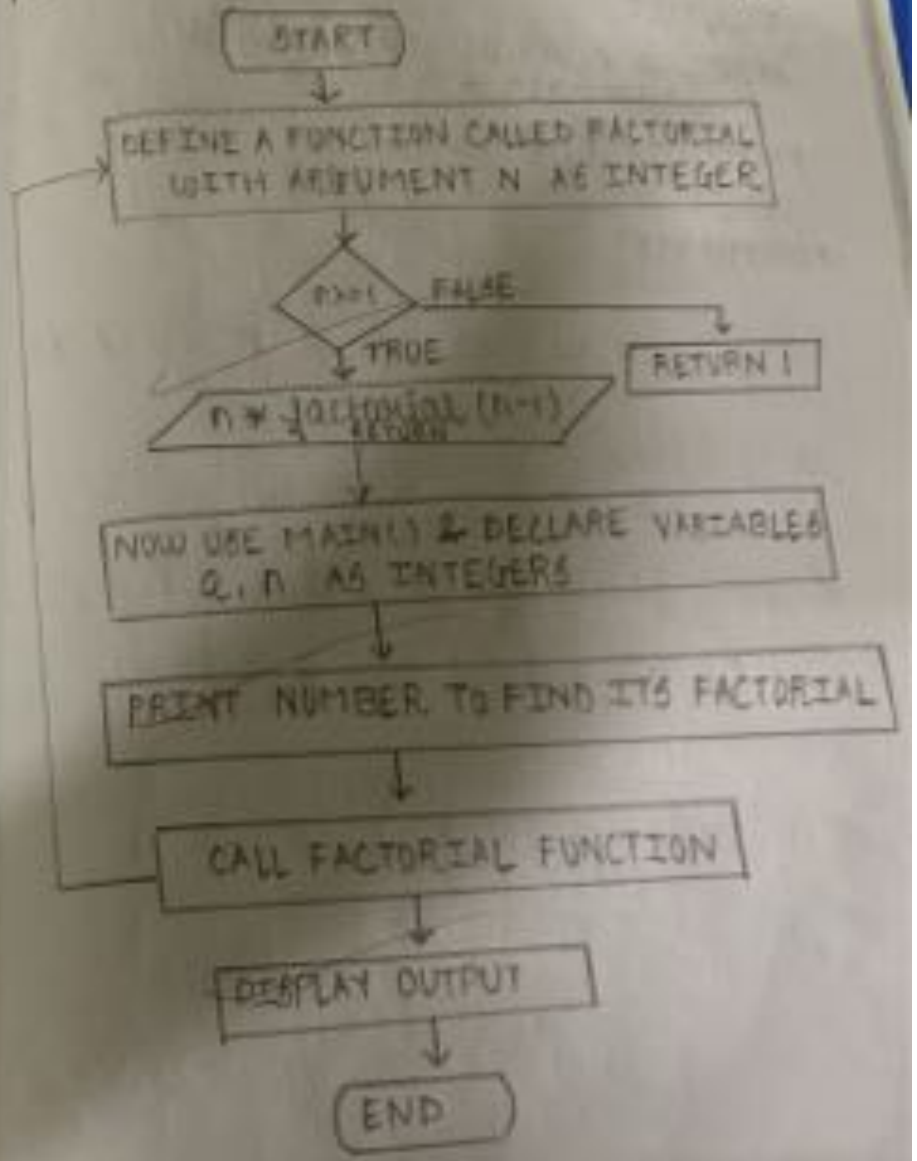
Step 3: Now use main () then declare a variables a, n as integers

Step 4: Now enter number to finds its factorial

Step 5: Now call the function factorial & display the answer.

Step 6: Terminate the program.

FLOWCHART:



START
DEFINE A FUNCTION CALLED FACTORIAL WITH ARGUMENT N AS INTEGER
n>1
FALSE
TRUE
RETURN 1
n * factorial (n-1) return
NOW USE MAIN() & DECLARE VARIABLES a, n AS INTEGERS
PRINT NUMBER TO FIND ITS FACTORIAL
CALL FACTORIAL FUNCTION
DISPLAY OUTPUT
END

Output:

```
Enter a number to find its
    factorial : 5
Factorial of 5 is 120
```

FLOWCHART:

Code:

```
#include <stdio.h>
#include <conio.h>
int factorial (int n)
{
        if (n>=1)
            return n * factorial(n-1);
        else
            return 1;
}
void main()
{
        int a, n;
        printf(" Enter a number to
              find its factorial: ");
        scanf("%d", &n);
        a = factorial(n);
        printf(" Factorial of %d is %d",
                n, a);
        getch();
}
```

Conclusion: Successfully executed the program to find factorial using recursion

Q.7 write a c program which shows the use of gets() function.

Algorithm:

step1: declare a variable name a array with size 50 with character datatype

step2: Now ask your name by user

step3: Now use gets() to read the input from user

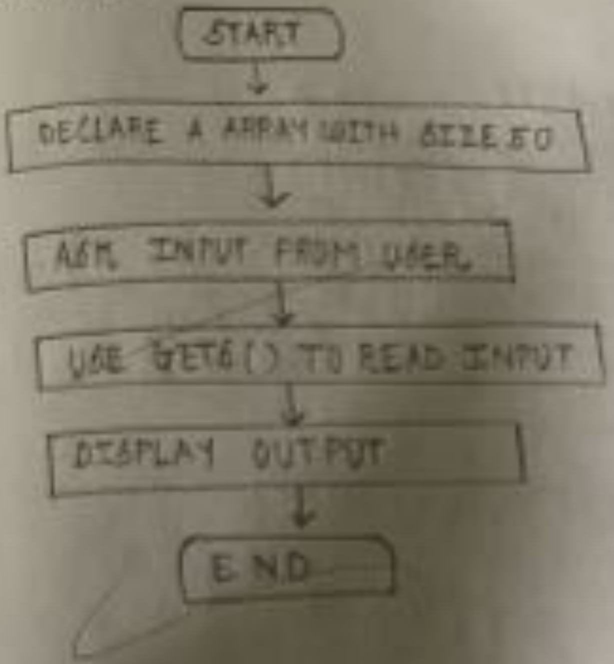
step4: Print your name on screen.

step5: Terminate the program

Program:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{       char name[50];
        printf("Enter your name:");
        gets(name);
```

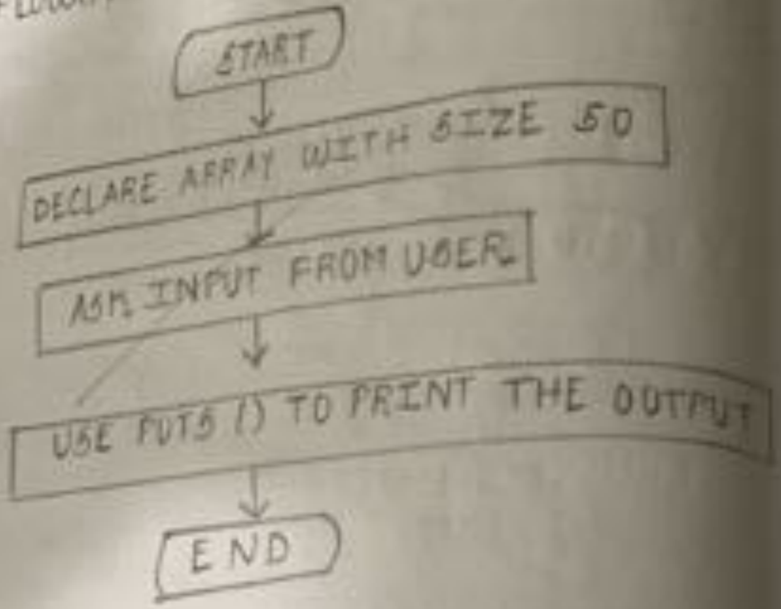
FLOWCHART



output:

```
Enter your name: Ankita
your name is Ankita
```

FLOWCHART:



```
printf("\n your name is %s",
                 name);
getch();
```

Conclusion: successfully executed the program.

8.] Write a C program to show the use of puts() function

Algorithm:

Step1: Declare a variable name as array with size 50 with character datatype

Step2: Now ask user to enter your name

Step3: Now use puts function to display your output

Step4: Terminate the program

code:

```
#include<string.h>
#include<stdio.h>
#include<conio.h>
void main()
{
        char name[50];
        printf("enter your name:");
        scanf("%s",&name);
        printf("\n your name is %s");
        puts(name);
        getch();
}
```

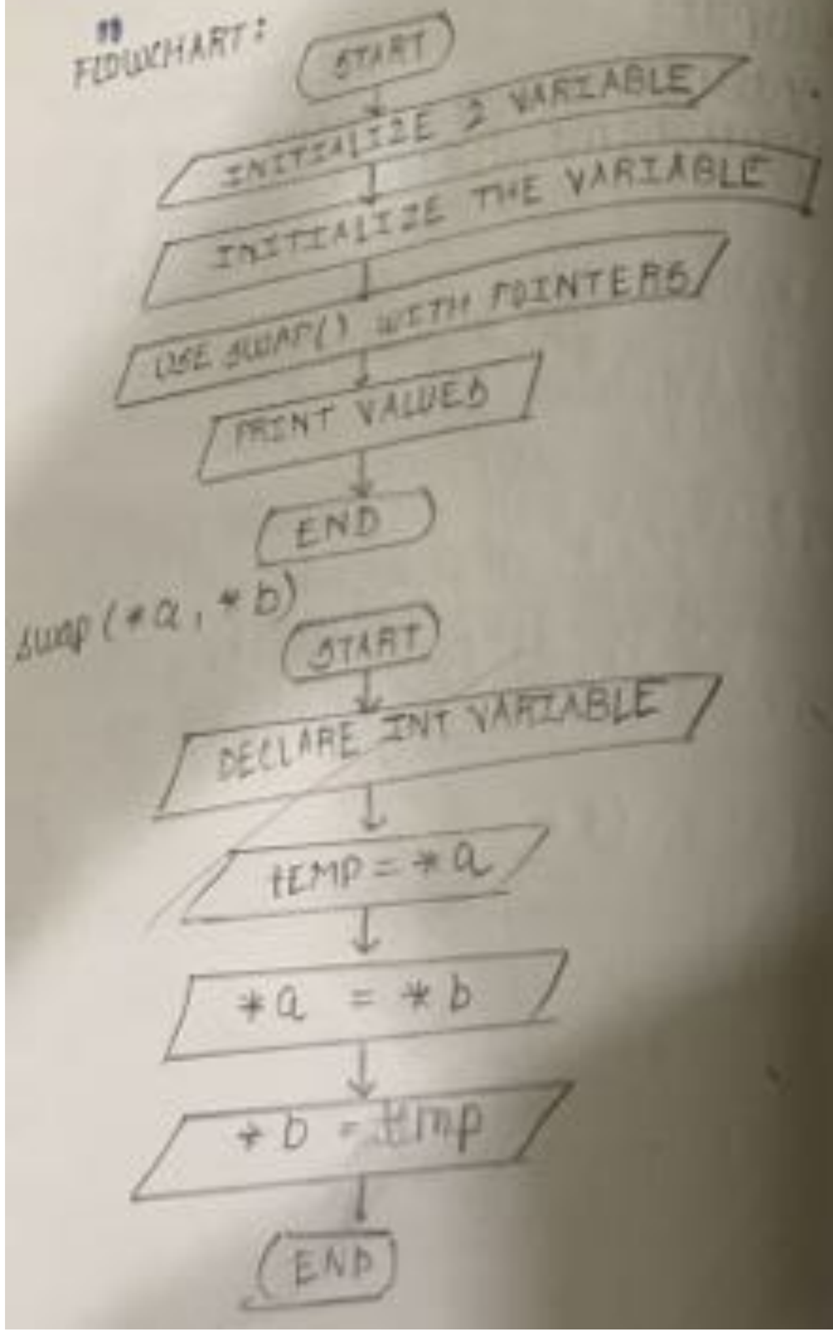
conclusion: successfully executed the program.

[illegible]

output:

```
enter your name : Ankita
your name is Ankita
```

FLOWCHART:



Practical 7

Aim: Program on pointers

Q] write a program to swap two numbers using pointers.

Algorithm:

Step1: Start the turbo C application

Step2: Declare a function prototype with 2 integer pointers as arguments before entering main()

Step3: Declare 2 variables and accept their values from user. Print the respective values using printf()

Step4: Place the address of the variable as arguments for the function

Step5: Print respective values of variables

Step6: Use the basic swapping algorithm in the function definition but instead using variables use pointers

#code:

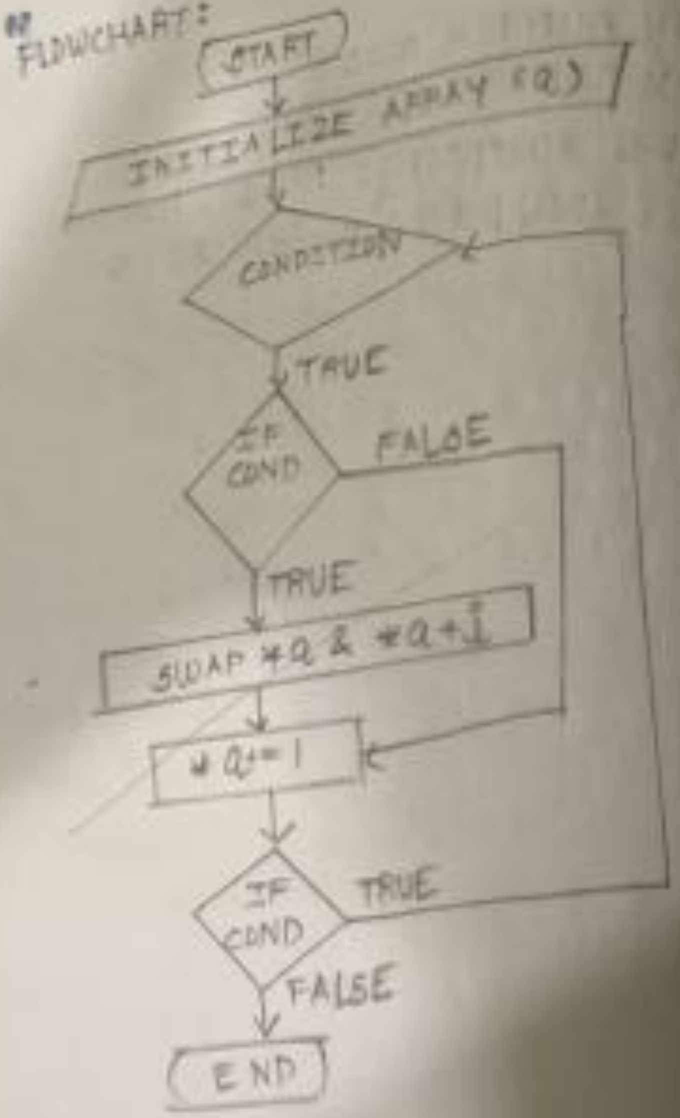
```
# include <stdio.h>
# include <conio.h>
void main ()
{
        int n1, n2;
        clrscr();
        printf ("Enter first number ");
        scanf ("%d" &n1);
        printf ("\n Enter second num-
          ber: ");
        scanf ("%d", &n2);
        printf ("\n before swapping
 : n1: %d, n2%d ", n1, n2);
    swap (&n1, &n2);
      printf (" After swapping
 : n1= %d , n2%d", n1, n2);
    getch();
}
int swap (int *a, int *b)
{
            int t;
            t = *a;
            *a = *b;
            *b = t;
}
```

Conclusion : successfully executed
        the program.

#output:

```
Enter number n1: 21
Enter number n2: 23
Before swapping: n1=21 , n2=23
After swapping : n1=23, n2=21
```

FLOWCHART:



Q.3] Write a C program to sort elements of array using pointers

Algorithm:

step1: Initialise an integer array and temp variable

step2: Run a nested loop of i=0 to len(a) & of j=0 to len(a)-1

step3: If *a > *a+1, swap the 2 values using basic swapping logic

step4: Print the swapped array

step5: Terminate the program

CODE:

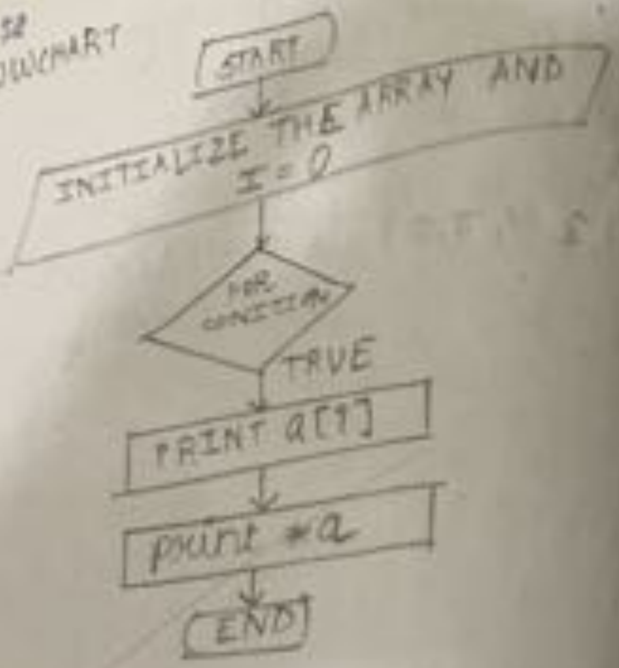
```
#include<stdio.h>
#include<conio.h>
void main()
{
        int x,y;
        clrscr();
        printf("Enter 2 nos to be
                swapped:");
        scanf("%d %d", &x, &y);
printf("\n the values before
    swapping are: %d & %d",
        x, y);

int a[10], temp, i, j;
clrscr();
for (i=0; i<10; i++)
{ for (j=0; j<10-i; j++)
  {
      if (*a > *a+i)
   {    temp = *a+i;
        *a+i = *a;
        *a   = temp;
   }
  }
}
printf("%d is the sorted
  array", a);
getch();
}
```

conclusion: successfully executed
    the program

output:

```
the elements in array:
1
6
7
8
2
{1,2,6,7,8} is sorted list of
                         array.
```

FLOWCHART



Q.9] write a c program for one dimensional array representation using pointers

Algorithm:

step1: start the turbo c application

step2: Initialize an integer array and a variable

step3: Run a for loop with i=0 to length of array

step4: Print the data of the array and the use pointer to print the address

step5: Terminate the program

CODE:

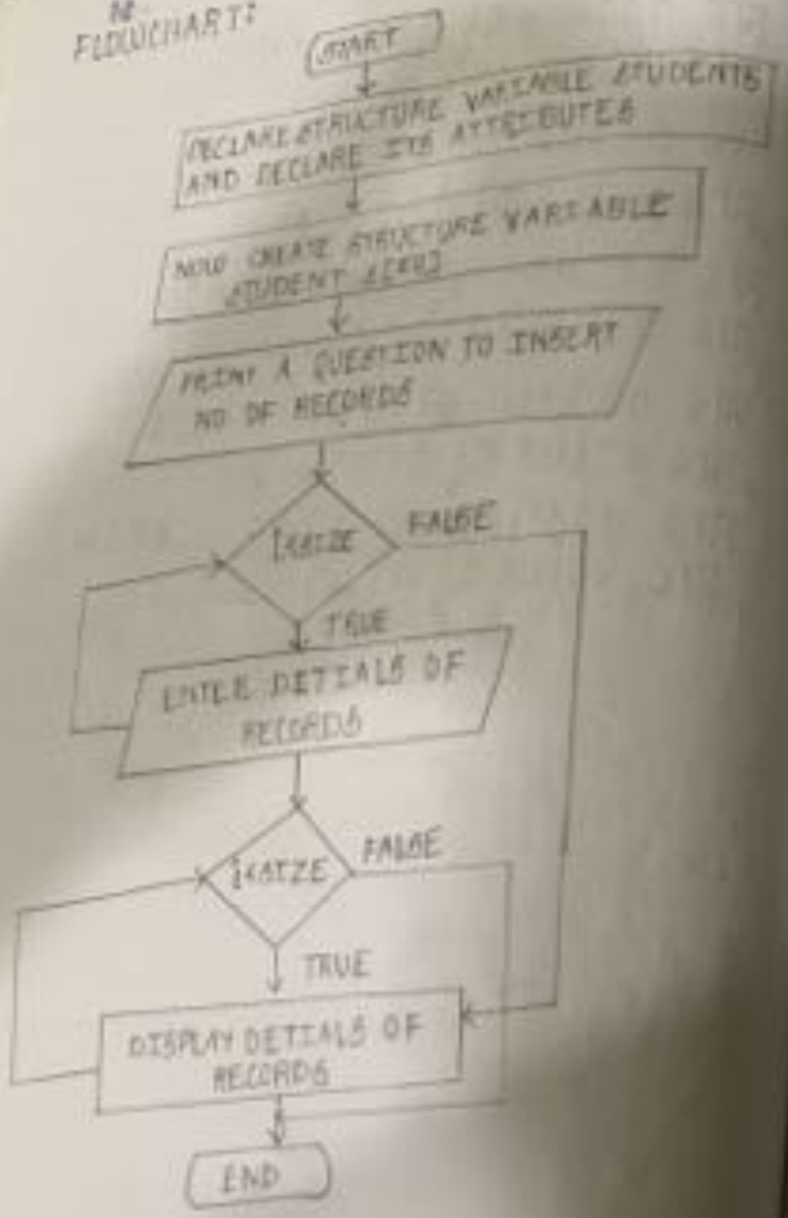
```
# include < stdio.h>
# include < conio.h>
void main()
{
    int a[5] = {7,9,4,8,2}
    int *ptr;
    int i=0;
    clrscr();
    ptr = &a[0];
    while(*ptr != '\0')
  {   printf("\n the address
      of a[%d] = %u", i, ptr);
      printf("\n the value of
      a[%d] = %d", i, *ptr);
      ptr++;
      i++;
  }
  getch();
}
```

conclusion: successfully executed
the program

output:

```
the address of a[0] = 65516
the value of a[0] = 7
the address of a[1] = 65518
the value of a[1] = 9
the address of a[2] = 65520
the value of a[2] = 4
the address of a[3] = 65522
the value of a[3] = 8
the address of a[4] = 65524
the value of a[4] = 2
```

FLOWCHART:


START
DECLARE STRUCTURE VARIABLE STUDENTS AND DECLARE ITS ATTRIBUTES
NOW CREATE STRUCTURE VARIABLE STUDENT S[80]
PRINT A QUESTION TO INSERT NO OF RECORDS
i<SIZE
FALSE
TRUE
ENTER DETAILS OF RECORDS
i<SIZE
FALSE
TRUE
DISPLAY DETAILS OF RECORDS
END

Practical 8

Aim: Programs on structures and unions

Q] Create a simple structure named student that holds the following variable ① Id ② CGPA ③ Name

Algorithm:

Step1: Start the turbo C application

Step2: Declare a structure variable student & in it also declare its attributes ie id as integer, CGPA as float and name as character datatype

Step3: Now create structure student s[80] as array and variable size & i as integer

Step4: Now print a question to ask how many record you user want to be inserted.

Step5: Now use for loop to enter the details of record step: Again we for loop to display the details of the records

```
code :
#include<stdio.h>
#include<conio.h>
void main()
{     struct student
   {      int id;
          float cgpa;
          char name[30];
   };
   struct student s[50];
   int size, i;
   printf ("\n How many records
you want to insert : ");
   scanf ("%d", &size);
   for (i=0; i<size; i++)
 {   printf ("Enter the ID: ");
     scanf ("%d", &s[i].id);
     printf ("\n enter the cgpa :");
     scanf ("%f", &s[i].cgpa);
     printf ("\n Enter Name :");
     scanf ("%s", &s[i].name);
 }
printf ("\n the student records\n\n");
printf ("\t ID   \t CGPA    \t NAME \n\n"
```

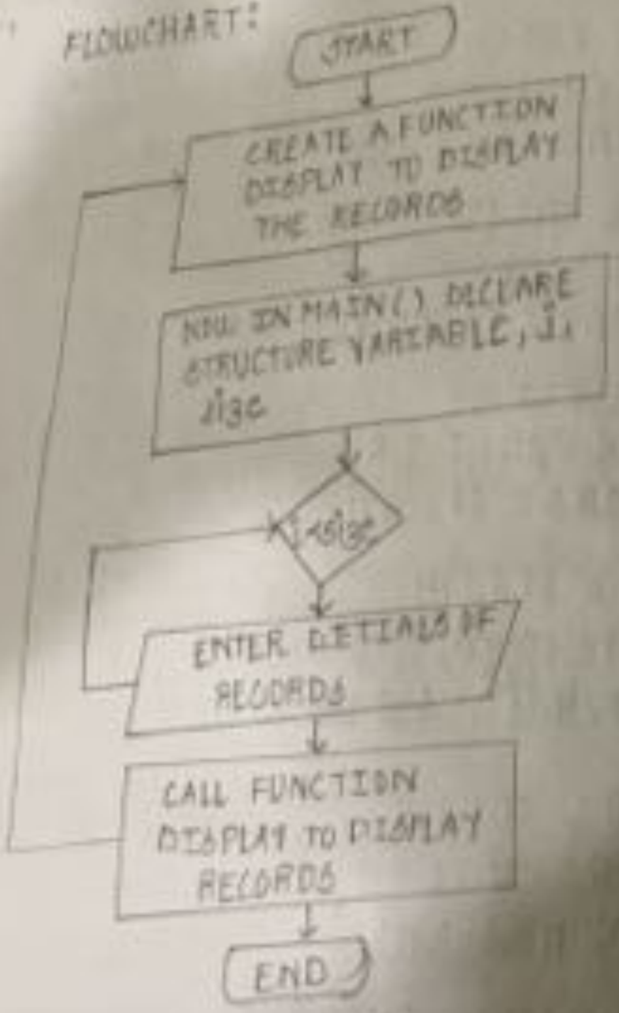
OUTPUT:

```
How many records you want to insert:
                                    5

Enter the ID: 101
Enter the cgpa: 9.3
Enter Name : Namrata

Enter the ID: 102
Enter the cgpa : 8.4
Enter Name : Dhanush

Enter the ID: 103
Enter the cgpa: 7.6
Enter Name : Dhruvi

Enter the ID: 104
Enter the cgpa : 10
Enter Name : Aakash

Enter the ID: 105
Enter the cgpa: 8.9
Enter Name: Neel

The student Record :

   ID          CGPA         NAME
   101         9.3          Namrata
   102         8.4          Dhanush
   103         7.6          Dhruvi
```

32

104	10	Aarsh
105	89	Neel

FLOWCHART:



START
CREATE A FUNCTION DISPLAY TO DISPLAY THE RECORDS
NOW IN MAIN() DECLARE STRUCTURE VARIABLE, i, size
i<size
ENTER DETAILS OF RECORDS
CALL FUNCTION DISPLAY TO DISPLAY RECORDS
END

33

```
    for(i=0; i<size; i++)
    {  printf("%d\t%d [illegible] \n",
         s[i].id, s[i].age, s[i].name);
    }
    getch();
}
```

Conclusion: Program executed successfully

Q3] Write a program which will demonstrate the use of function & structure.

Algorithm:

Step1: Start turbo C application

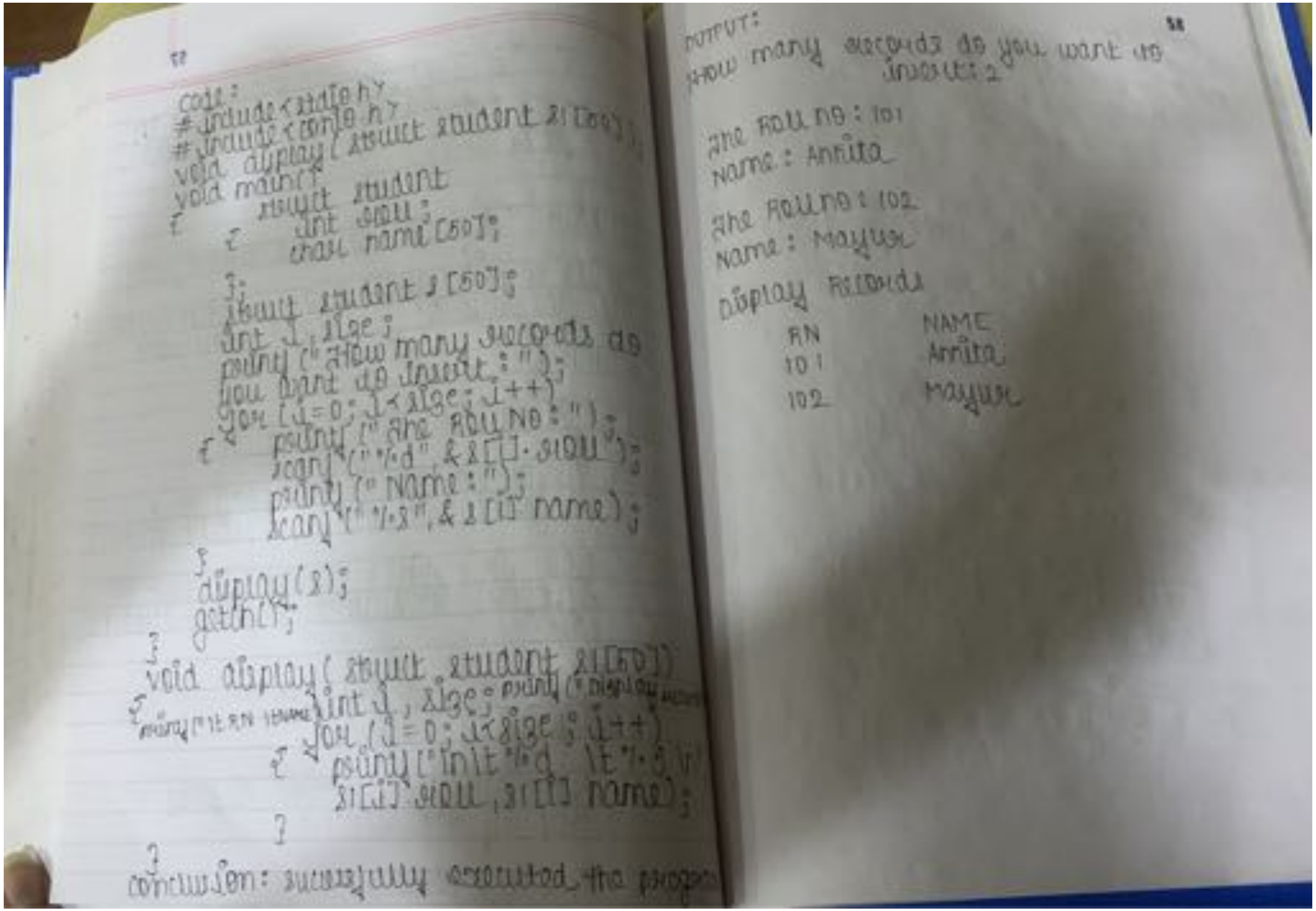
Step2: Declare a function display with argument as struct student s[50])

Step3: Now in main() declare i, size as integer, structure variable student s[50];

Step4: Now use for loop to insert the details of records.

Step5: Now call the function display to display the records.

Step6: End the program.



code:

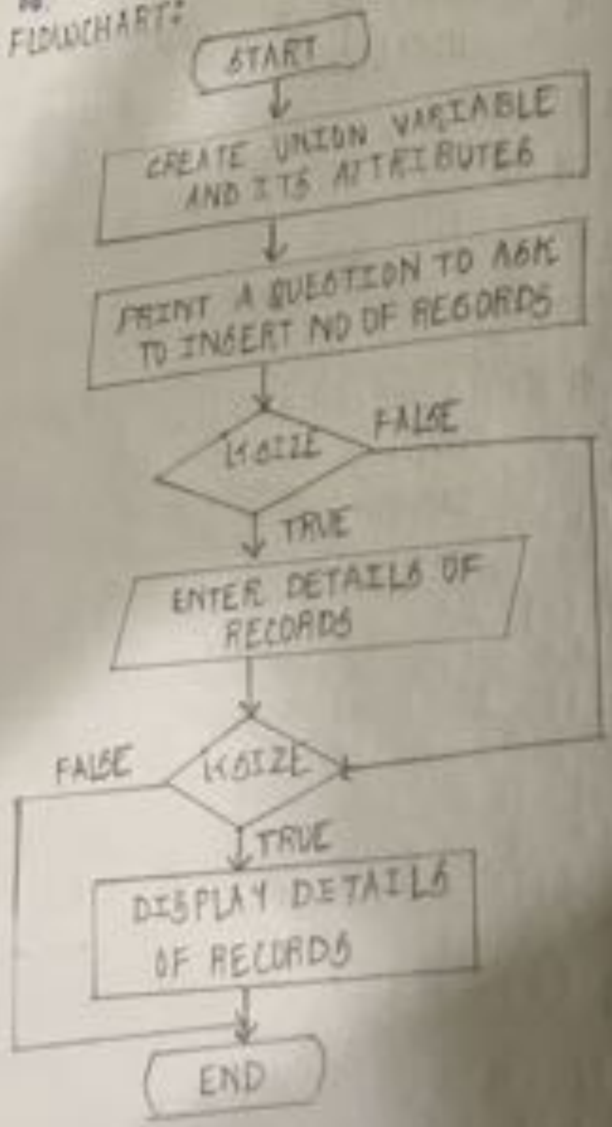
```
#include<stdio.h>
#include<conio.h>
void display(struct student s1[50]);
void main()
{     struct student
   {     int roll;
         char name[50];
   };
   struct student s[50];
   int i, size;
   printf("How many records do
   you want to insert:");
   for(i=0; i<size; i++)
 {     printf("The Roll No:");
       scanf("%d", &s[i].roll);
       printf("Name:");
       scanf("%s", &s[i].name);
   }
   display(s);
   getch();
}
void display(struct student s1[50])
{   int i, size; printf("Display records
printf("\t RN \tName     for(i=0; i<size; i++)
      {   printf("\n\t %d \t%s\n",
             s1[i].roll, s1[i].name);
      }
}
```

conclusion: successfully executed the progr

OUTPUT:

```
How many records do you want to
            insert:2

The Roll no: 101
Name: Anita

The Roll no: 102
Name: Mayur

display Records
      RN          NAME
      101         Anita
      102         Mayur
```

FLOWCHART:


START
CREATE UNION VARIABLE AND ITS ATTRIBUTES
PRINT A QUESTION TO ASK TO INSERT NO OF RECORDS
I<SIZE
FALSE
TRUE
ENTER DETAILS OF RECORDS
I<SIZE
FALSE
TRUE
DISPLAY DETAILS OF RECORDS
END

Q.3) Create union to store data of a student in the form of roll no, stud name, div, percentage, contact no. Insert data & print the same.

Algorithm:

Step1: Start turbo C application

Step2: Declare union variable student & in it we declare roll no & contact no as integer and array, percent as float and div and stud name as character datatype

Step3: Now create union student s[60] & declare i and size as integer

Step4: Now use for loop to insert the details of records

Step5: Now use for loop again to insert display details of records

Step6: End the program.

Code:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    union student
    {
        int roll_no, contact_no[10];
        float percent;
        char div, name[30];
    };
    union student s[50]; int i, size;
    printf("How many records do
    you want to insert : ");
    scanf("%d", &size);
    for(i=0; i<size; i++)
    {
        printf("\nRoll No : ");
        scanf("%d", &s[i].roll_no);
        printf("\nContact No : ");
        scanf("%d", &s[i].contact);
        printf("\nName : ");
        scanf("%s", &s[i].name);
        printf("\nDivision : ");
        scanf("%c", &s[i].div);
        printf("\nPercentage : ");
        scanf("%f", &s[i].percent);
    }
    printf("\n\t RN \t Name \t Div
        \tContact NO \t percentage \n
```

output :

```
How many records do you want to
        insert : 3

Roll NO : 1001
contact NO : 9999799999
Name : Naruk
Division : A
Percentage : 88.67

Roll NO : 1002
contact NO : 8888899999
Name : Harsh
Division : A
Percentage : 90.18%

Roll NO : 1003
contact NO : 7777785658
Name : Priya
Division : B
Percentage : 76.52
```

RN	NAME	DIV	Contact No	Percentage
101	Nakul	A	9999999999	88.67
102	Harsh	A	8888899999	90.18
103	Riya	B	7777755668	76.62

```
  for (i=0; i<3; i++)
{
    printf("\n\t%d \t%s \t%c,
          \t%d \t%f",
  s[i].roll_no , s[i].name , s[i].div,
  s[i].contact_no , s[i].percentage);
}
  getch();
}
```

Conclusion: Program executed successfully

Practical 9
Aim: Programs on string manipulation

Q] Write a program to copy one string into another string

Algorithm:

Step1: Declare a character array and pointer *t. I as integer

Step2: Ask user to enter a string

Step3: Now use t=1 to copy base address of string

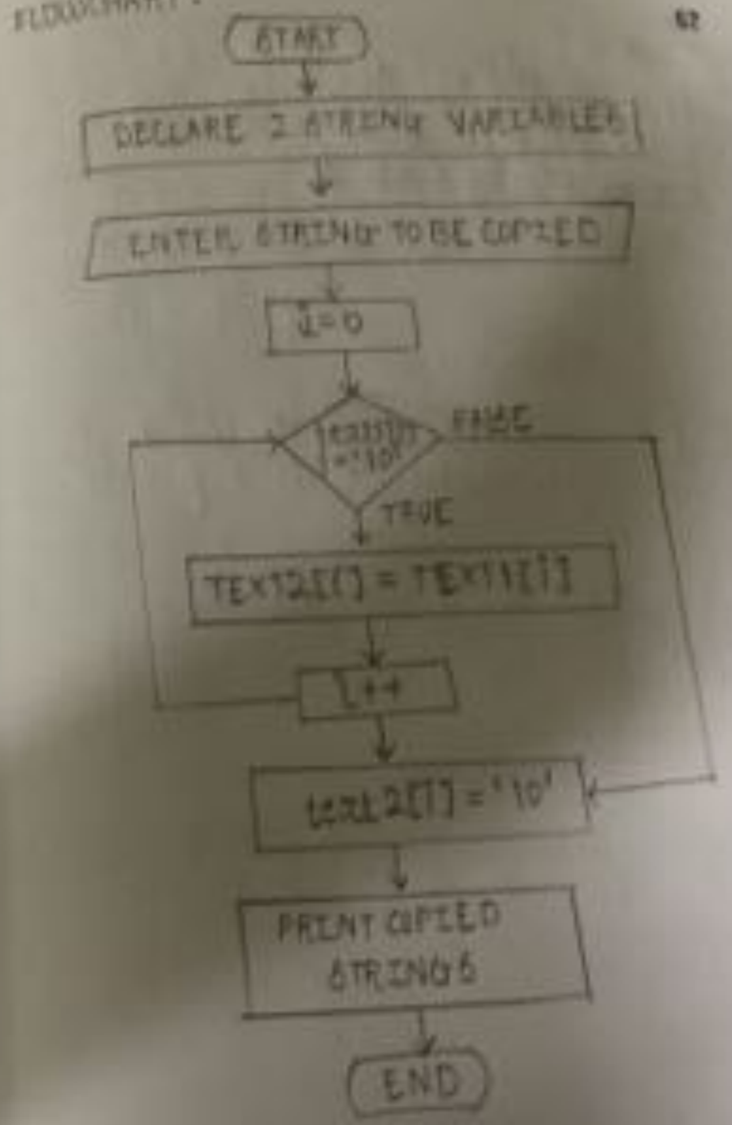
Step3: use while loop to check null character in character array to copy the two arrays

Step4: Now check null character for 2 string variable to [illegible] the loop

Step5: Print copy copied strings

Step6: End the program

FLOWCHART:



```
Output:
Enter a string: abcd
First string: abcd
Second string : abcd
```

Code:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        char text1[60];
        char text2[60];
        int i;
        i=0;
        printf(" Enter a string:");
        scanf("%s", &text1);
        while (text1[i] != '\0')
        {
                text2[i] = text1[i];
                i++;
        }
        text2[i]='\0';
        printf(" First string = %s \n", text1);
        printf(" Second string = %s", text2);
        getch();
}
```

Conclusion: Program executed successfully

Q.5 Write a program which will demonstrate the string library functions

Algorithm:

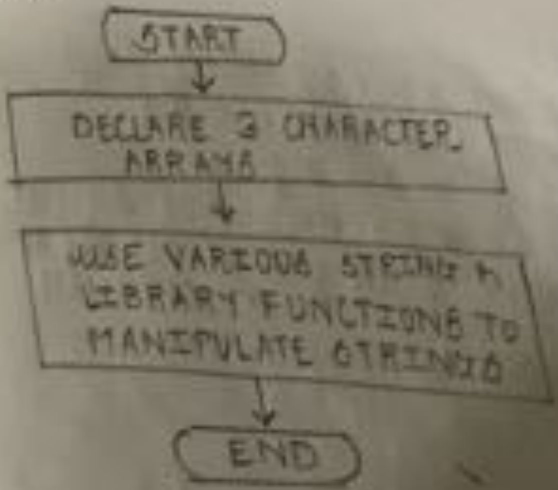
Step1: Start the turbo c application

Step2: Declare two character array variables

Step3: Use various string.h library functions to manipulate the two strings.

Step4: End the program

FLOWCHART:


START
DECLARE 3 CHARACTER ARRAYS
USE VARIOUS STRING.h LIBRARY FUNCTIONS TO MANIPULATE STRINGS
END

```
Output:
 Enter first string : Hello
 Enter second string : world
 Enter second string : 5
length of first string : not equal
string 1 & string 2 are
output after string concatenation:
                         Helloworld

text1 : world
```

Code :

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
        char text1[30], text2[30], text3[60];
        printf("Enter first string : ");
        scanf("%s", & text1);
        printf("Enter second string : ");
        scanf("%s", & text2);
        printf("\n length of string 1 : %d",
               strlen(text1));
      if (strcmp(text1, text2) == 0)
    {     printf(" string 1 & string 2 are
           equal");
    }
    else
    {   printf(" string 1 & string 2 are
          not equal");
    }
    text3 = strcat(s1, s2);
   printf(" output after string concate
      nation : ", text3);
   text strcpy (text1, text2)
   printf("  text1 : ", text1);
   getch();
}
```

Conclusion : Program successfully executed.

Q.3] Write a program which displays the length of a string without using string function

Algorithm:

Step1: Start turbo c application

Step2: Declare a character array, *t & len=0 as integer,

Step3: Now enter a string

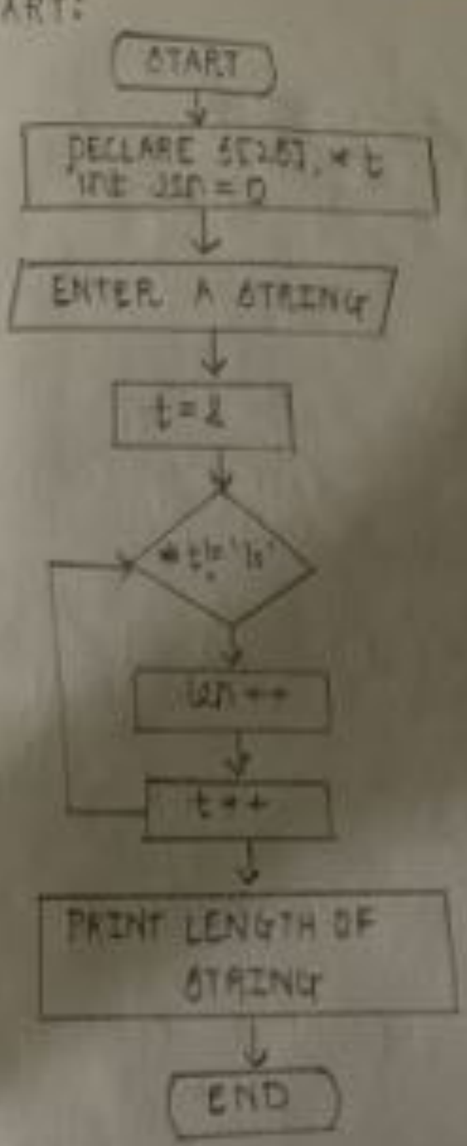
Step4: Use t=s to copy base address of string

Step5: Use while loop to count length of string

Step6: Print length of string

Step7: End the program

FLOWCHART:



#output:

```
Enter a string: Hello
length of string is 5
```

#CODE:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        char s[25], *t;
        int len = 0;
        printf("\n Enter a string :");
        scanf("%s", &s);
        t=s;
       while (*t!='\0')
      {
            len++;
            t++;
      }
      printf(" length of a string is %d",
                              len);
     getch();
}
```

conclusion: Program executed successfully

Practical 10:

Aim: Programs on basic file operations

Q.I Program for file open, file read & file close.

Algorithm:

Step1: Start turbo c application

Step2: Declare FILE * fptr and num as Integer.

Step3: Check if there is a existing file by using if else statement if the pointer is NULL print that there is error opening the file & exit the program

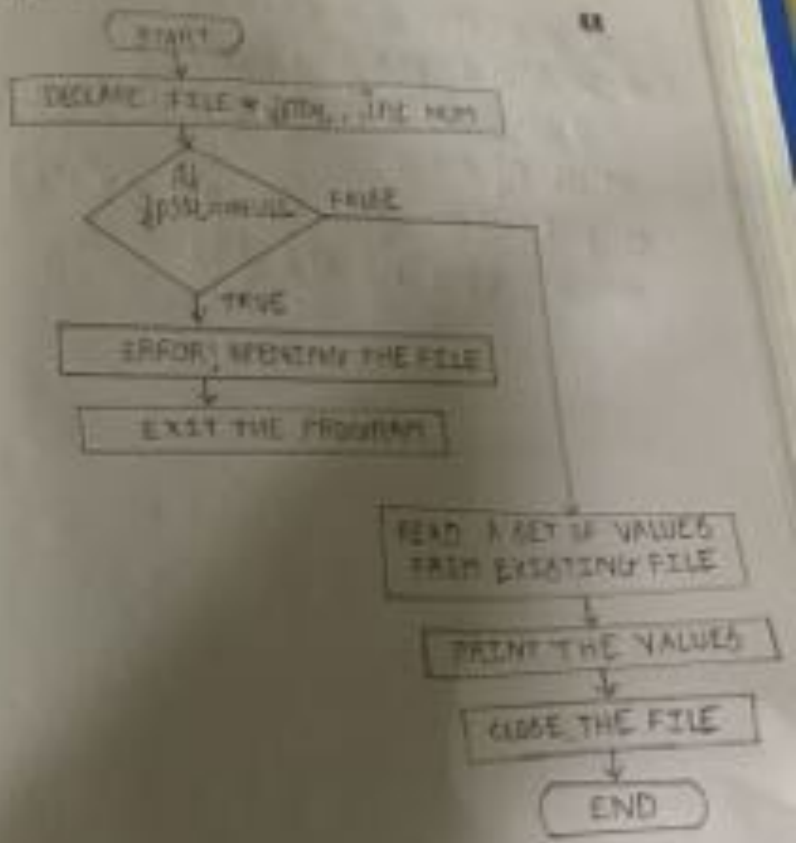
Step4: Now use fscanf to read a set of data values from a existing file

Step5: Now print the content of file

Step6: Now use fclose to close the file

Step7: End the program

FLOWCHART:



Output : (when file exists)
# contents of program.txt :
There are 3 languages in our syllabus
value of n=3
output : (when file doesn't exists)
Error! opening the file

code :

```
#include<stdio.h>
#include<conio.h>
void main()
{
            int num;
            FILE *fptr;
            clrscr();
            if ((fptr = fopen("C:\\program.txt",
                  "r")) == NULL)
            {       printf("Error! opening
                    file");
                    exit(1);
            }
            fscanf(fptr,"%d",&num);
            printf("value of n=%d",num);
            fclose(fptr);
            getch();
}
```

Conclusion : Program executed successfully.

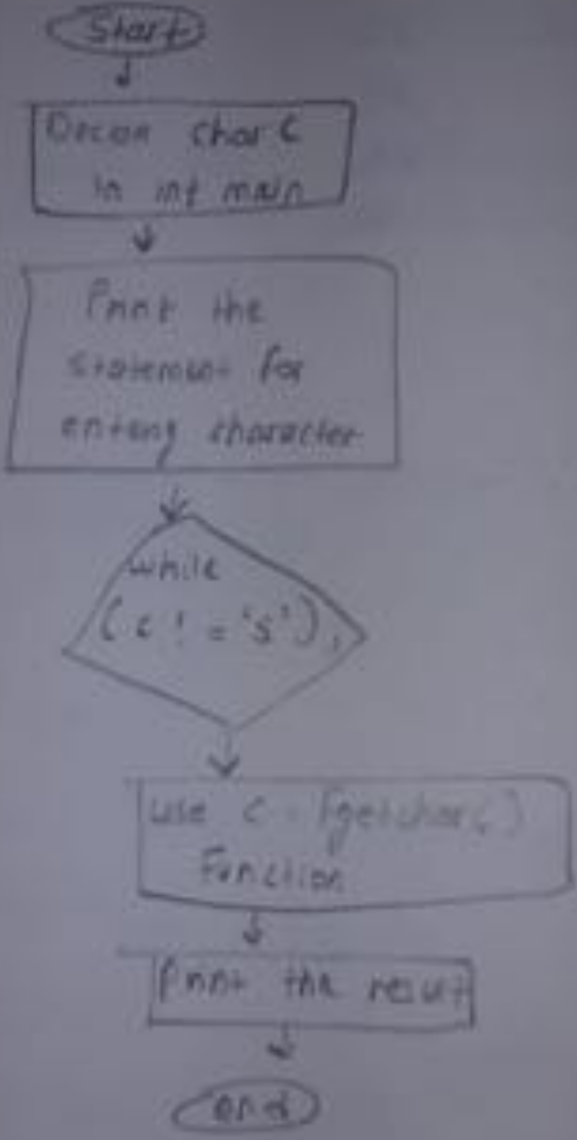
Aim : WAP for fgetc(), fgetchar(), fputchar() function

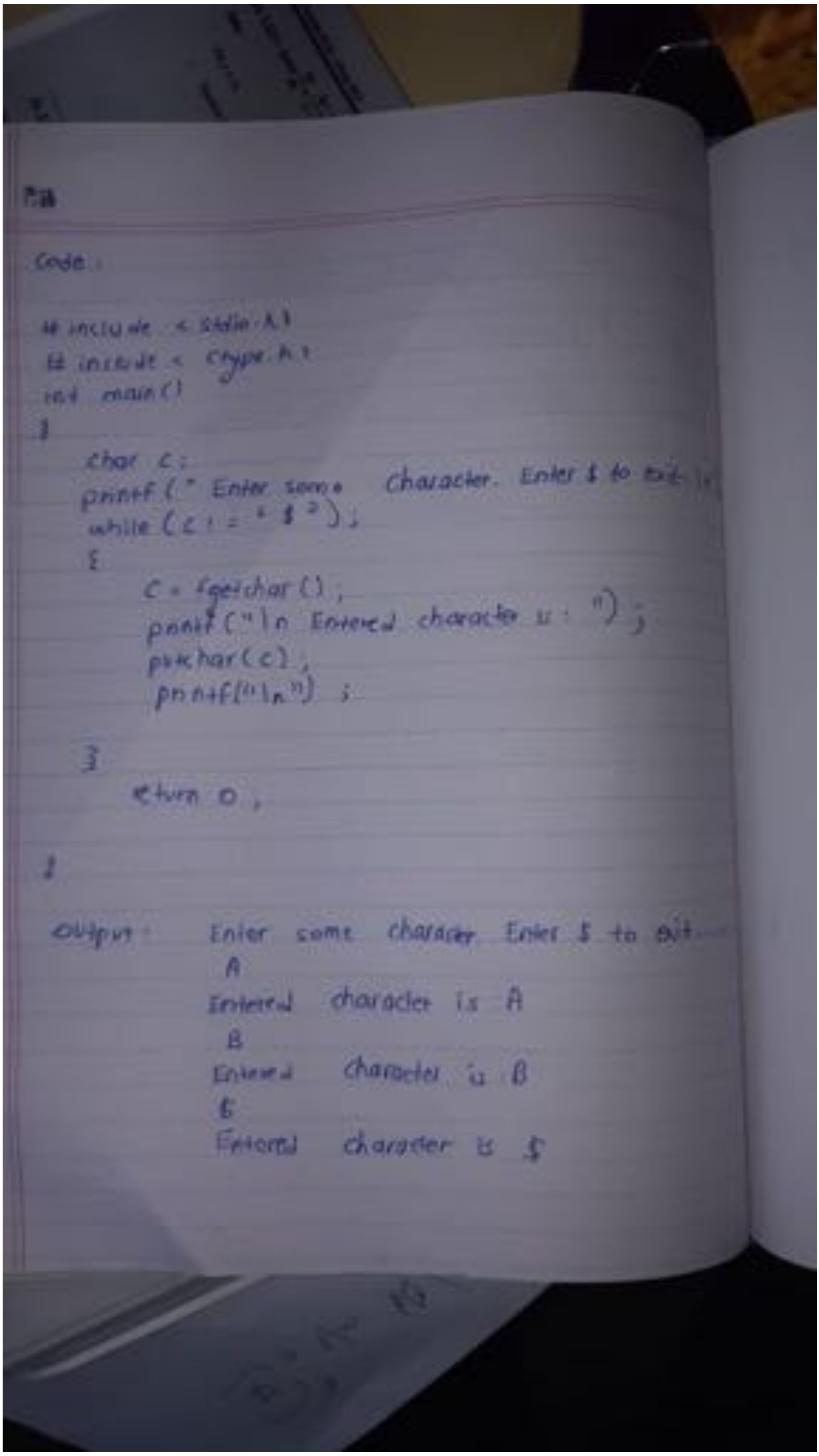
Algorith / Description

[illegible] - Fgetchar is a file handling function

- It is used to read a single character From keyboard input.

1a


Start
Declar char c
in int main
Print the
statement for
entering character
while
(c != 's'),
use c = fgetchar()
function
Print the result
end



P.18

Code :

```
# include < stdio.h >
# include < ctype.h >
int main()
{
    char c;
    printf(" Enter some character. Enter $ to exit \
    while (c != '$');
    {
        c = fgetchar();
        printf("\n Entered character is : ");
        putchar(c);
        printf("\n");

    }
      return 0;

}
```

Output :

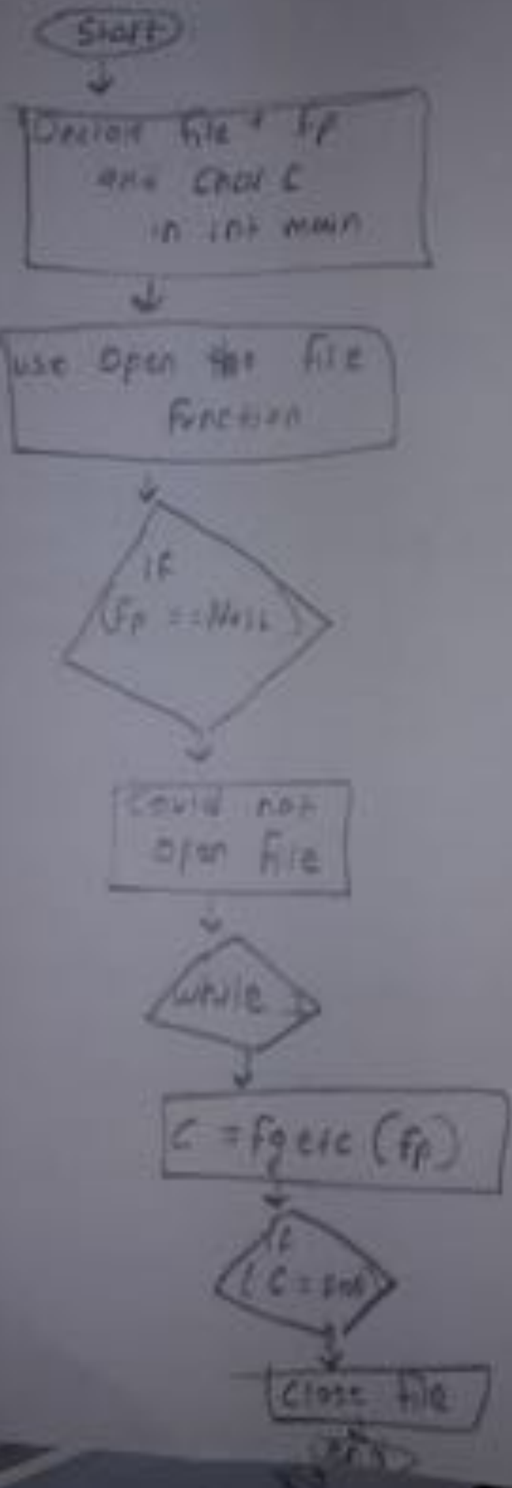
```
Enter some character. Enter $ to exit
 A
Entered character is A
 B
Entered character is B
 $
Entered character is $
```

fgetc() → Used to read a character from a file.
Reads single character at a time.
In a program we use fgetc() function
fgetc(fp);
where

Fp = file pointer

Code :

```
include< stdio.h >
int main ()
{
 file * fp,
 char c ;
 printf ("opening file test.c in read mode");
 fp = fopen ("\n test.c", "r");
 if ( fp == NULL)
 {
  printf(" Could not open file test.c");
  return 1 ;
 }
 printf(" Reading the file test.c");
 while (1)
 {
  c = fgetc( fp);
  if (c = eof)
   break ;
  printf ("%c",c) ;
}
```


Start
Declare file * fp and char c in int main
use open the file function
if (fp == NULL)
could not open file
while
c = fgetc (fp)
if (c == EOF)
close file
end

[illegible]

```
printf ("closing file test.c");
fclose ( fp);
return 0;
}
```

output:

output:-

```
Opening the file test.c in read mode
  Reading the file test.c
  Hi, How are you?
   Closing the file test.c.
```