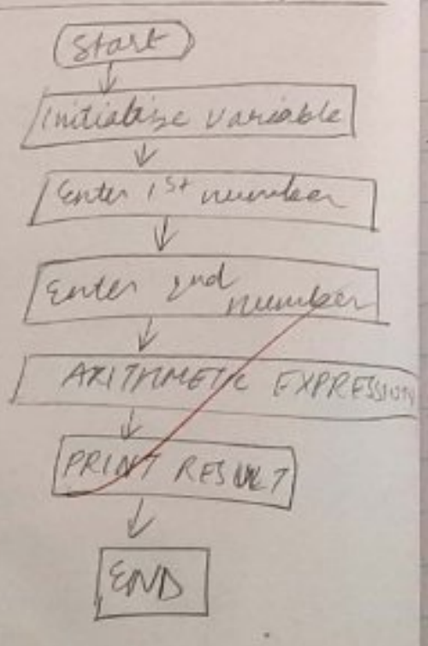
35 INPUT:

```
#include <stdio.h>
#include <conio.h>
void main ()
{
    int roll_no;
    char name[20], mobile_no [10];
    float percentage;
    clrscr();
    printf("enter student's roll_no :\n");
    scanf ("%d", &roll_no);
    printf("enter student's name :\n");
    scanf ("%s", &name);
    printf("enter student's mobile_no :\n");
    scanf ("%s", &mobile_no);
    printf ("enter student's percentage :\n");
    scanf ("%f", &percentage);
    printf ("student's name : %s\n", name);
    printf ("student's mobile_no : %s\n",mobile_no);
    printf ("student's percentage : %.f\n", percentage);
    printf ("student's roll_no : %d\n", roll_no);
    getch();
}
```

OUTPUT:

```
Enter student's roll_no : 1896
enter student's name : sakshi
enter student's mobile_no: 75544459
enter student's percentage: 88.55
student's name : Sakshi
student's mobile_no: 75544459
student's percentage: 88.55
student's roll-no : 1896
```



29

Practical - 01

Aim: Write a C-Program to understand the basic datatypes and I/O.

- Algorithm / Theory:

S1: Declare a variable, name, roll no as integer, also declare name, mobile no. as characters & percentage: as float.

S2: Use printf function to print questions for the user in order to give input.

S3: Use scanf function to read user's input & store in its allocated memory.

S4: Again, use printf function to display the output.

- CONCLUSION: The given program gives user an idea about how built-in datatypes work in c and also about how user can give input & display output.

Nakshi

Scanned with CamScanner

PRACTICAL-02

Q1. (A) Write a program on operators & expression.

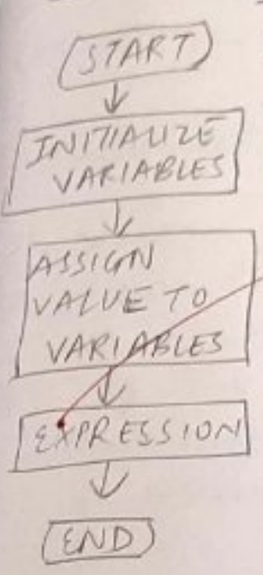
Algorithm:

S1: Declare a variable name for 1st & 2nd no. as integers.
S2: Use scanf fn. to recieve input from user.
S3: Now to add 2 nos. given by user, use the expression : num1 + num2.
S4: Now to subtract 2 nos. given by user, use expression : num1 - num2.
S5: Again use expression num1 * num2 if user wishes to multiply 2 inputs.
S6: Again use expression num1/num2 if user wishes to divide the 2 inputs.
S7: Use expression ~~no~~ Now use printf fn to display the output.

C/C++ Program for DYNAMIC CALCULATOR :—

```
include <stdio.h>
include <conio.h>
void main ()
{
    int x, y, z;
    clrscr();
    printf("Return any two numbers\n");
    scanf("%d", "%d", &x, &y);
    z = x+y;
    printf("sum of two numbers = %d \n", z);
    z = z-y;
    printf("subtraction of two numbers = %d\n",
    z);
    z = x/y;
    printf("Division of two numbers = %d\n",
    z);
    z = x % y;
    printf("Modulus of two numbers = %d\n",
    z);
    z = x * y;
    printf("Multiplication of two numbers =
    %d\n", z);
    getch();
}
```

*FLOWCHART



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
Modulus of 3 and 3 is 0.0000
```


Scanned with CamScanner

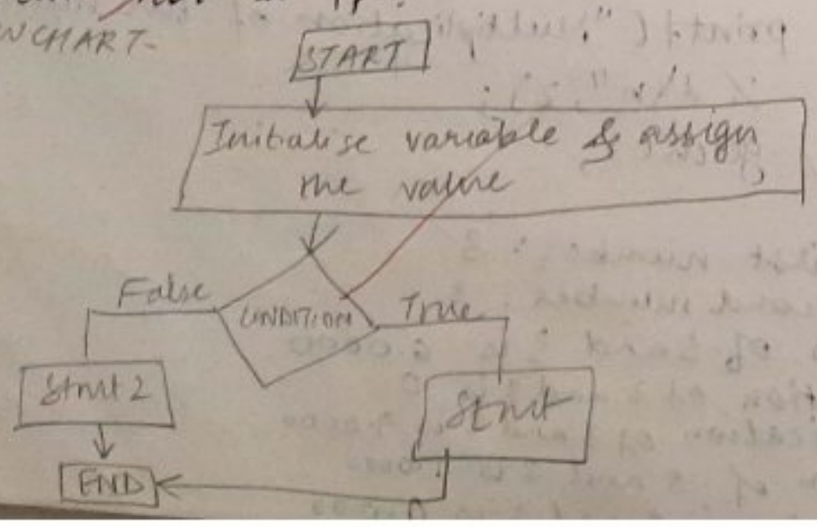
(A) Program for IF statement —

```
#include <stdio.h>
#include <conio.h>
void main()
{
        int i=10;
        clrscr();
        if (i>15)
{
        printf("10 is less than 15\n");
}
        else()
        printf("I am not in if \n");
        getch();
}
```

output:

"I am not in if.

FLOWCHART-



Q1. Write a program on decision statements.

Theory:

(a) Write a program in C to explain IF statement.

Algorithm:

S1: Declare a variable as integer and assign it 20.
S2: Now, to compare whether 20 is greater than 15, use if statement.
S3: If the condition is true, printf that 20 is less than 15 & If condition is false skip the if statement and print "I am not in if"

(b) Write a program in C to explain if else statement.

Algorithm:

S1: Declare a variable as integer and assign its value as 20.

S2: Now to compare the given value if its gre or not, use if else conditional statement.

Scanned with CamScanner

S3: If condition is true then print "20 is less than 15. or if condition i is false then print "20 is greater than 15".

(C) Write a program in C to explain nested "if statement.

Algorithm:

S1: Declare a variable as integer and assign value i.e. 20.

S2: Now use nested if logic to compare if given no. is greates or not.

S3: If first condition is true then go to second condition if second condition is also true then print that 20 is greater than 12 & 15.

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

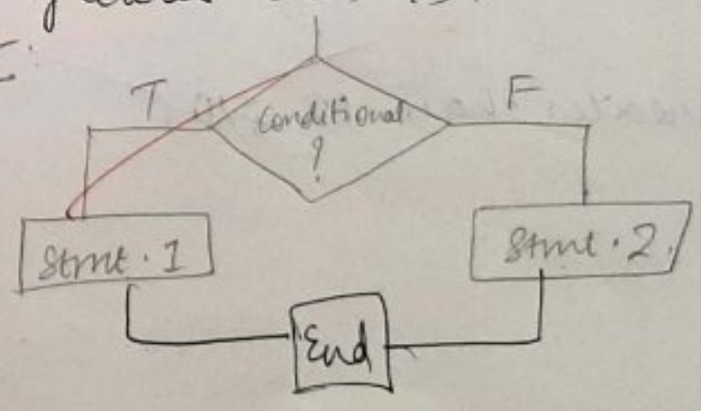
(B) Program for IF ELSE statement :-

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

output : :

20 is greater than 15.

Flowchart:



Scanned with CamScanner

(c) Program for NESTED IF :-

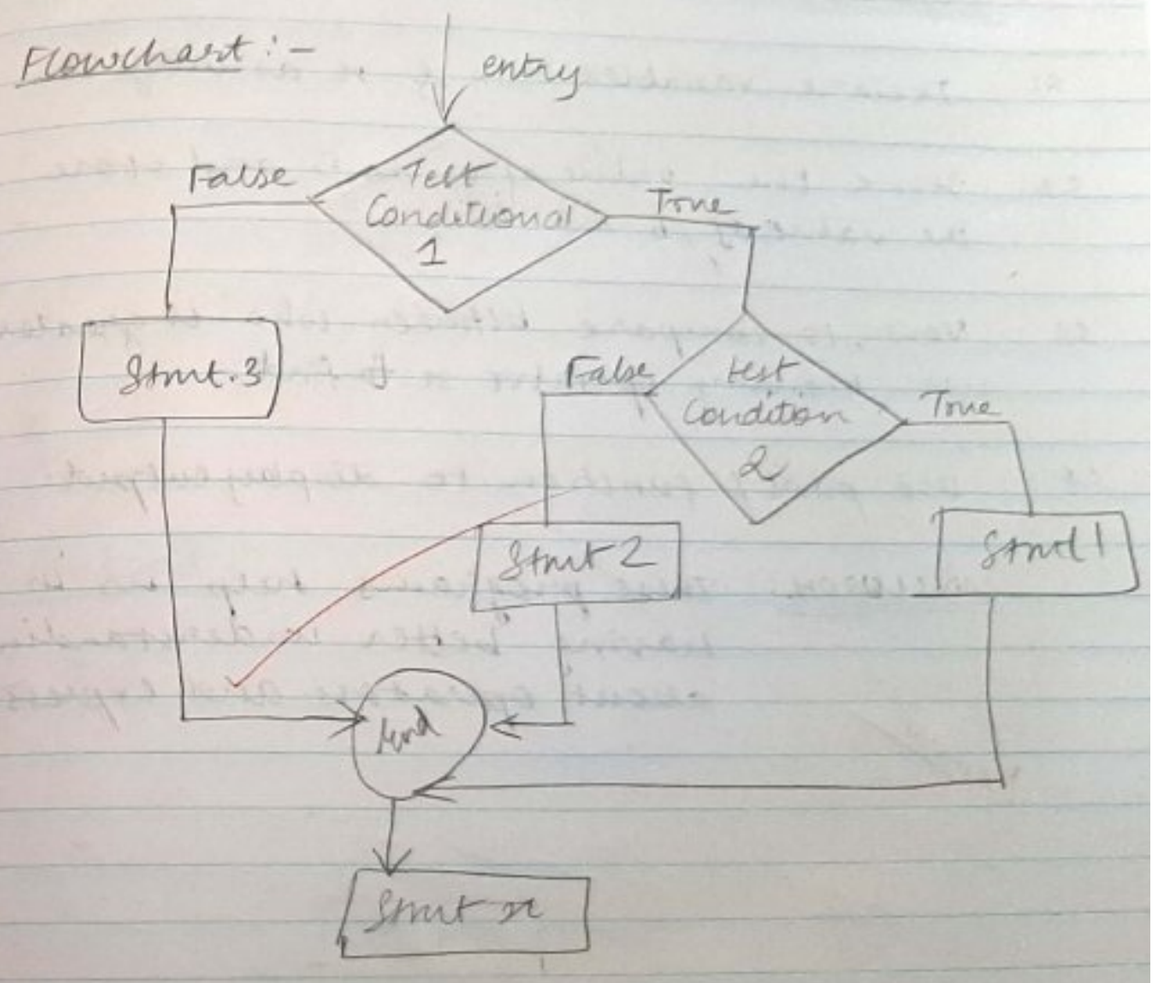
```
#include <stdio.h>
#include <conio.h>
void main()
{
        int=10;
        clrscr();
        if(i==10);
{
        if(i<15)
        printf("i is smaller than 15\n");
        if(i<12)
        printf("i is smaller than 15\n");
        else
        printf("i is greater than 15\n");
}
        getch();
}
```

output:

20 is greater than 12 & 10.

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

Flowchart :-



Scanned with CamScanner

CS

Practical-02

(B) Write a program in c to explain Teornary operator :

Algorithm:

S1: Declare variables a, b & x as integers.

S2: Store the value of a as 5 and store the value of b as 15.

S3: Now, to compare between who is greater use Ternary operator x to find.

S4: Use printf function to display output.

CONCLUSION: These programs help us in having better understanding about operators and expressions.

Nirmala

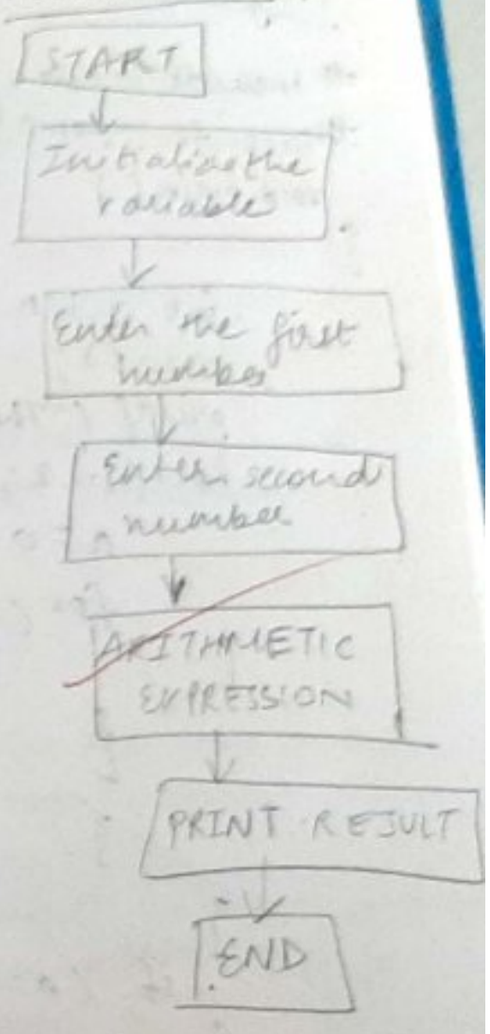
(B) code for TERNARY OPERATOR :- 31

```
// teornary 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 :



Scanned with CamScanner

Program:

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


PRACTICAL-04

Aim: To display prime numbers using for loop.

ALGORITHM:

S1: Initialize three variables out of which two are loop variables and are count variable.

S2: Initialize a for loop from 2-50. Set the count variable to 0.

S3: Put initialize loop within the loop in S2 that goes from 2 to first loop variable % 2.

S4: Use the if conditional statement to check whether 1st loop variable % 2nd variable == 0) if true increment count variable by 1.

S5: Come out of second loop and check whether the count variable is 0 if true, print the number.

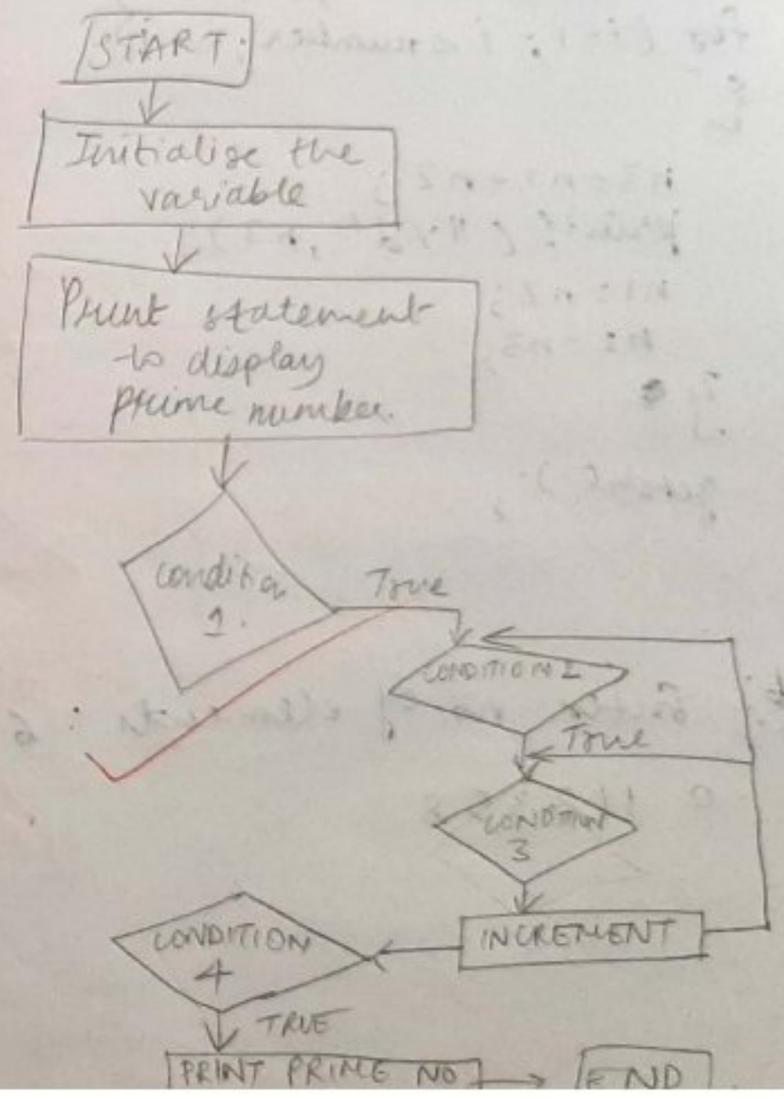
Scanned with CamScanner

SC: Terminate the program:

CONCLUSION: Prime numbers were displayed using for loop.

*OUTPUT: The prime numbers are: 2 3 5 7 11 13 15 17 19.

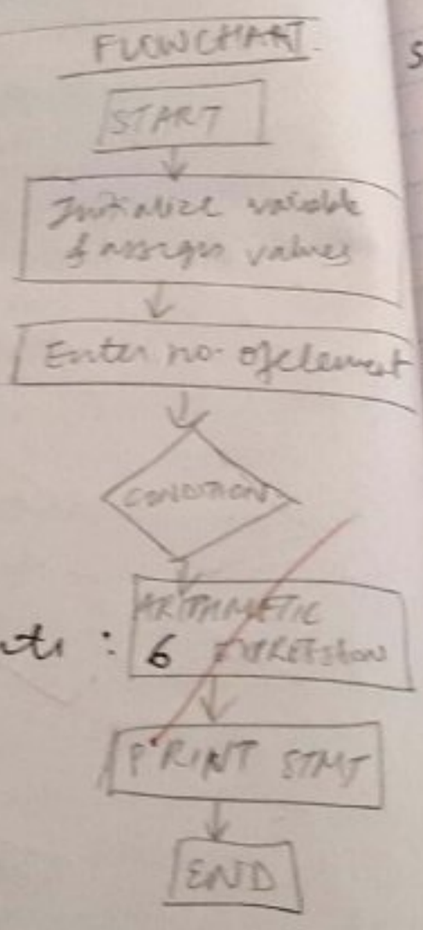
*FLOWCHART



Scanned with CamScanner

Program :-

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int n1=0, n2=1, n3, i, number;
    printf("no. of elements \n");
    scanf("%d", &number);
    printf("%d %d", n1, n2);
    for(i=1; i<number; ++i)
    {
        n3=n1+n2;
        printf("%d", n3);
        n1=n2;
        n2=n3;
    }
    getche();
}
```



Output: Enter no. of elements : 6

0 1 1 2 3 5 8

(b) **Aim:** Write a program on Fibonacci series.

Algorithm:

S1: Start the Turbo C program.

S2: Declare variables n1, n2, n3, i, number.

S3: Initialize the variable n1=0, n2=1, number=0.

S4: Enter the number of terms of fibonacci series to be printed

S5: Print first terms of series as n1=0, n2=1.

S6: Use for loop as per following step.

n3 = n1 + n2

n1 = n2

n2 = n3

Increase the value of i element each time by 1.

S7: Print value of numbers.

S8: End.

CONCLUSION: Thus, we have successfully execu fibonacci series on turbo C.

Scanned with CamScanner

(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:

S1: Start Turbo C program.

S2: Declare variable rows, i, j, number=1.

S3: Display number rows.

S4: Enter the for loop as i=1, i<=rows, i++.

S5: Now create nested - for loop as j=1, j<=i, ++j.

S6: Display the number as per user enter the sequence from i=1.

S7: Increment the no. from 1.

S8: Display space.

*Program:


```
#include <stdio.h>
#include <conio.h>
void main()
{
    int n1=0, n2=1, n3, i, number;
    printf("Enter no. of elements \n");
    scanf("%d
    int n1=0, r, i, j;
    clrscr();
    printf("Enter no. of rows");
    scanf("%d", &r);
    for(i=0; i<=r; i++)
    {
        for(j=0; j<=i; j++)
        {
            n1++;
            printf("%d", n1);
        }
        printf("\n");
    }
    getch();
}.
```

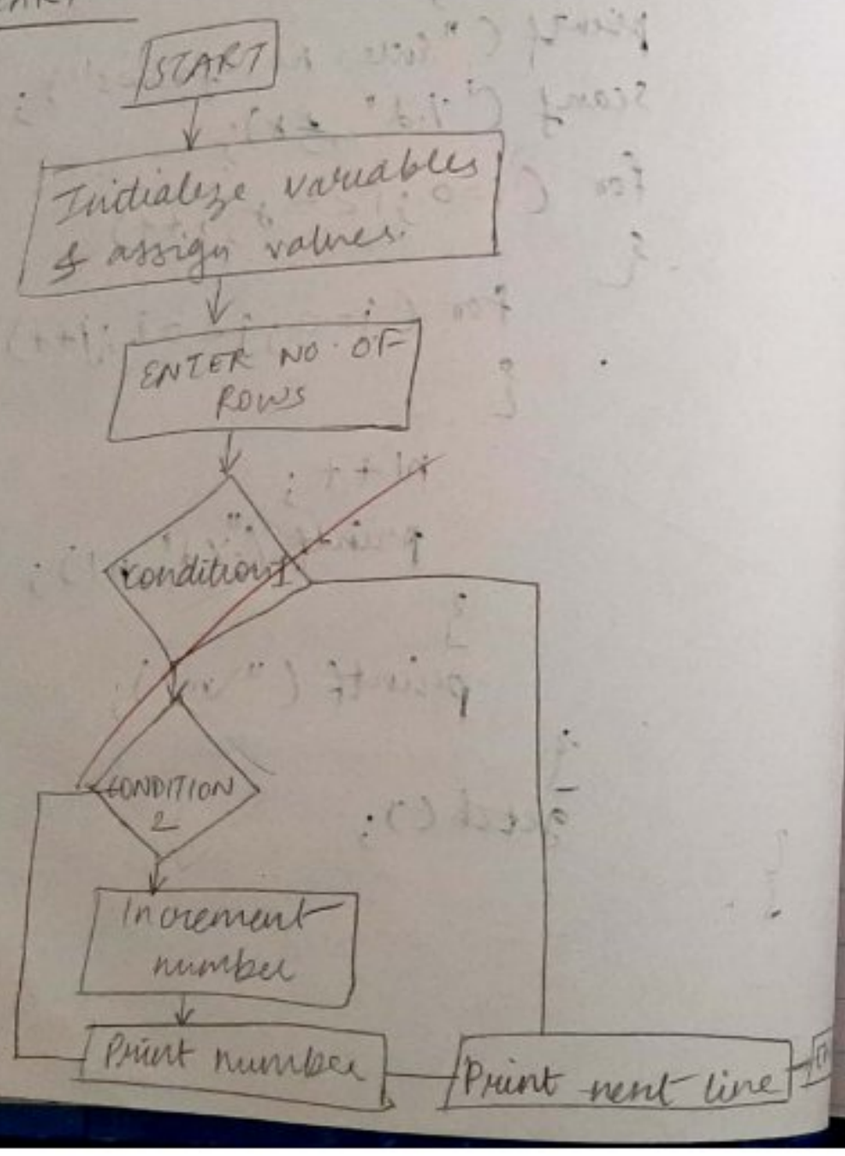

Scanned with CamScanner

Output:

Enter the number of rows: 3

1
2 3
4 5 6
7 8 9 10

FLOWCHART



59: End.

Conclusion: Thus, we have successfully executed given expression on Turbo C using for loop. Nested

Rhmadi

Scanned with CamScanner

PRACTICAL 08

(a) Title: WAP to find largest no. among the array.

* Algorithm:

S1. Start the Turbo C.

S2. Declare variable i & int. array a[10].

S3: Enter for loop at i=0, i<10 and use value of a[i] in i<10. Exit for loop.

S4: Enter for loop at i=0, i<10, use if condition statement to check if a[0] < a[i] is true, put a[0] = a[i].

S5: Run the above for loop for i<10, exist the loop.

S6: Terminate the loop.

Conclusion: Hence, we have successfully learnt to find the largest number among the array.

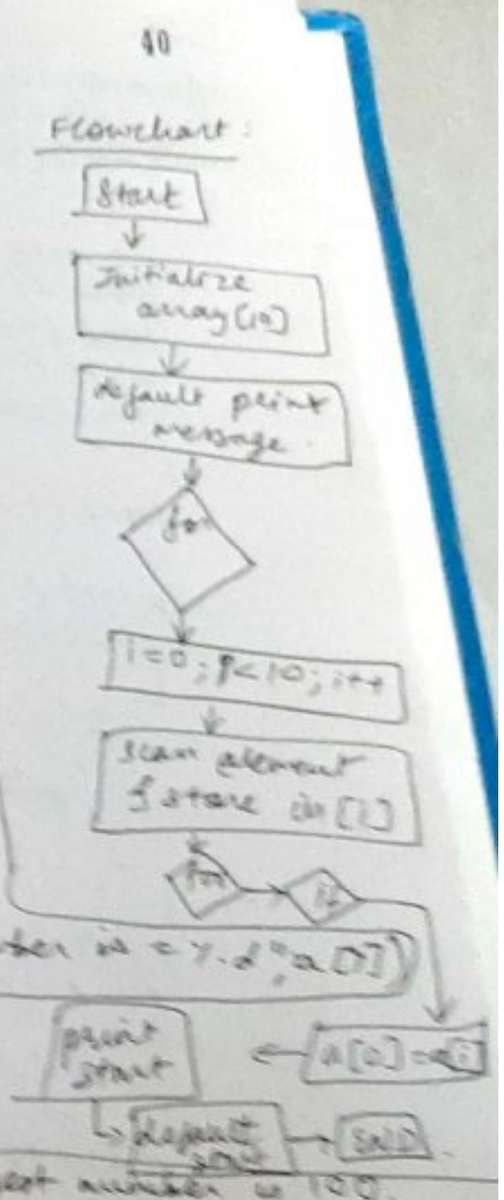
```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[10], i;
    clrscr();
    printf("Enter element of int");
    for (i=0; i<10; i++)
    {
      scanf("%d", &a[i])
    }
    for (i=0); i<10; i++)
    {
      if(a[0] < a[i])
      {
        a[0] = a[i];
      }
    }
    printf("The largest number is = %d", a[0])
    getch();
}
```

output:

```
12  21
23  12  22
2   55  100
    3
```

40

Flowchart:



Scanned with CamScanner

- CODE:

```
#include <stdio.h>
#include<conio.h>
void main()
{
   int array [100], i, num;
   printf("Enter the size of array\n");
   scanf ("%d", &num);
   printf("Enter the elements in array\n");
   for (i=0 ; i<num ;i++)
  {
  scanf ("%d",&array[i]);
  }
  printf ("Enter even no. in array");
   for (i=0;i<num;i++)
   {
     if (array[i] %. 2 ==0)
     {
       printf("%d", "array[i]);
     }
  }
  printf("\nodd no. in the array are");
  for (i=0; i<num ; i++)
  { if (array[i] %. 2 != 0)
  {
    printf("%d","array[i]);
  }
  }
   getch();
}.
```

Output:

```
Enter the size of array
4
Enter the elements of array
5
3
8
1
even no. → 8
Odd no. → 1 3 5
```

(2) WAP to print number of odd & even number in the array.

* Algorithm:

S1: Create an array. Take its size from user & define an element using loop.

S2: Display size of array from the user.

S3: Display element of array entered by user.

S4: Take the initiator in for loop using which all the elements of array are existing.

S5: Display even no. from the array using for loop. if (array[i] %. 2 == 0)

S6: Display odd no. → if (array[i] %2 != 0)

S7: Close / Terminate Turbo C.

CONCLUSION: Hence, we have learnt to find the odd and even number in the array.

Scanned with CamScanner

(3) Aim: WAP to find the sum ~~of~~ average of element in an array.

* Algorithm:

S1: Start Turbo C application.

S2: Declare int variable n, i. Initialize num[100], sum = 0.0, avg.

S3: Using for loop at i=0; i<n; i++. Give print message and increment by 1.

S4: Declare sum variable and store it by adding num[i]

S5: Average is sum divided by n.

S6: Give print statement for average and sum.

S7: Terminate the program.

CONCLUSION: Thus, we have executed the program successfully.

Ahmadi

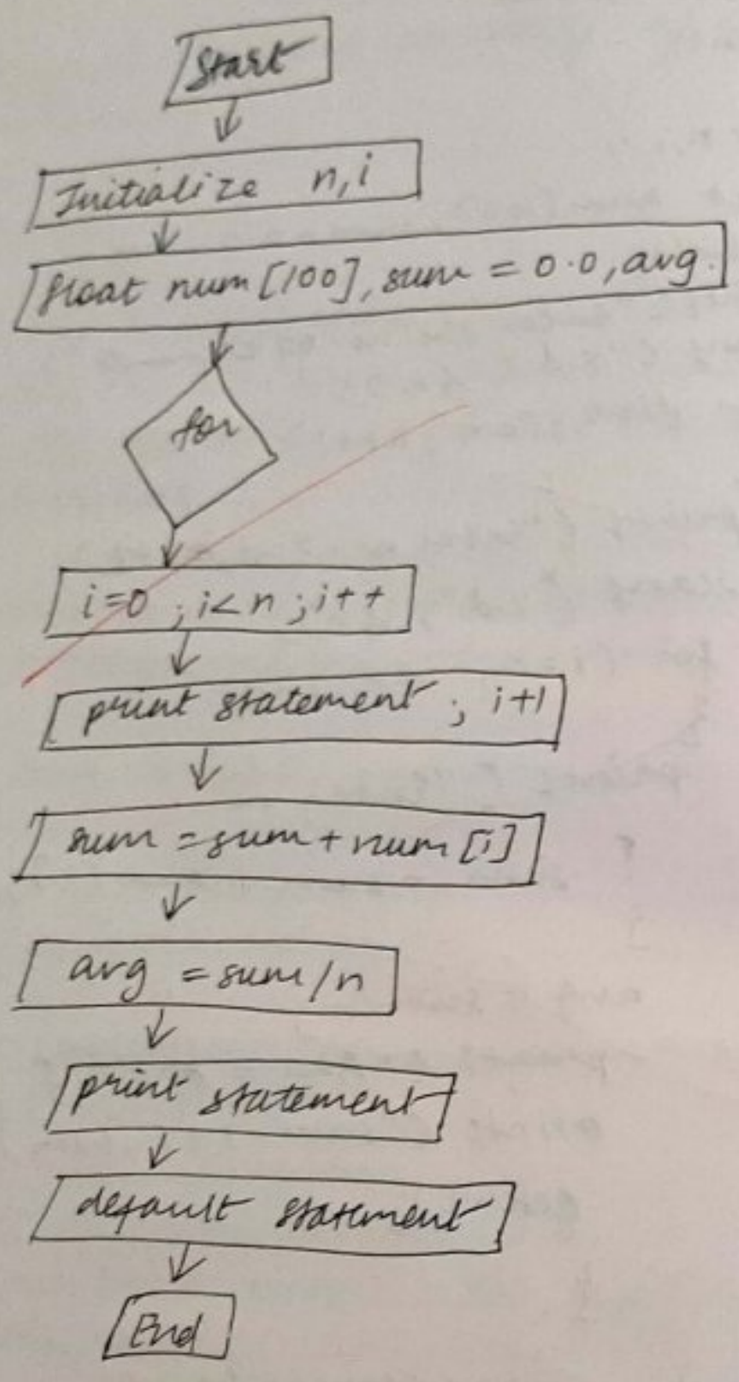
CODE:

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int n,i;
    float num[100], sum=0.0, avg;
    clrscr();
    printf("Enter the no. of elements");
    scanf("%d", &n);
    for(i=0;i<n;i++)
    {
    printf("Enter no. %d", i+1);
    scanf("%d", &n);
    for(i=0; i<n; i++)
    {
    printf("Enter no. .
    { sum = sum + sum[i];
    }
    avg = sum/n;
    printf("Avg = %.2f", avg);
    printf("sum = %.2f", sum);
    getch();
}.
```

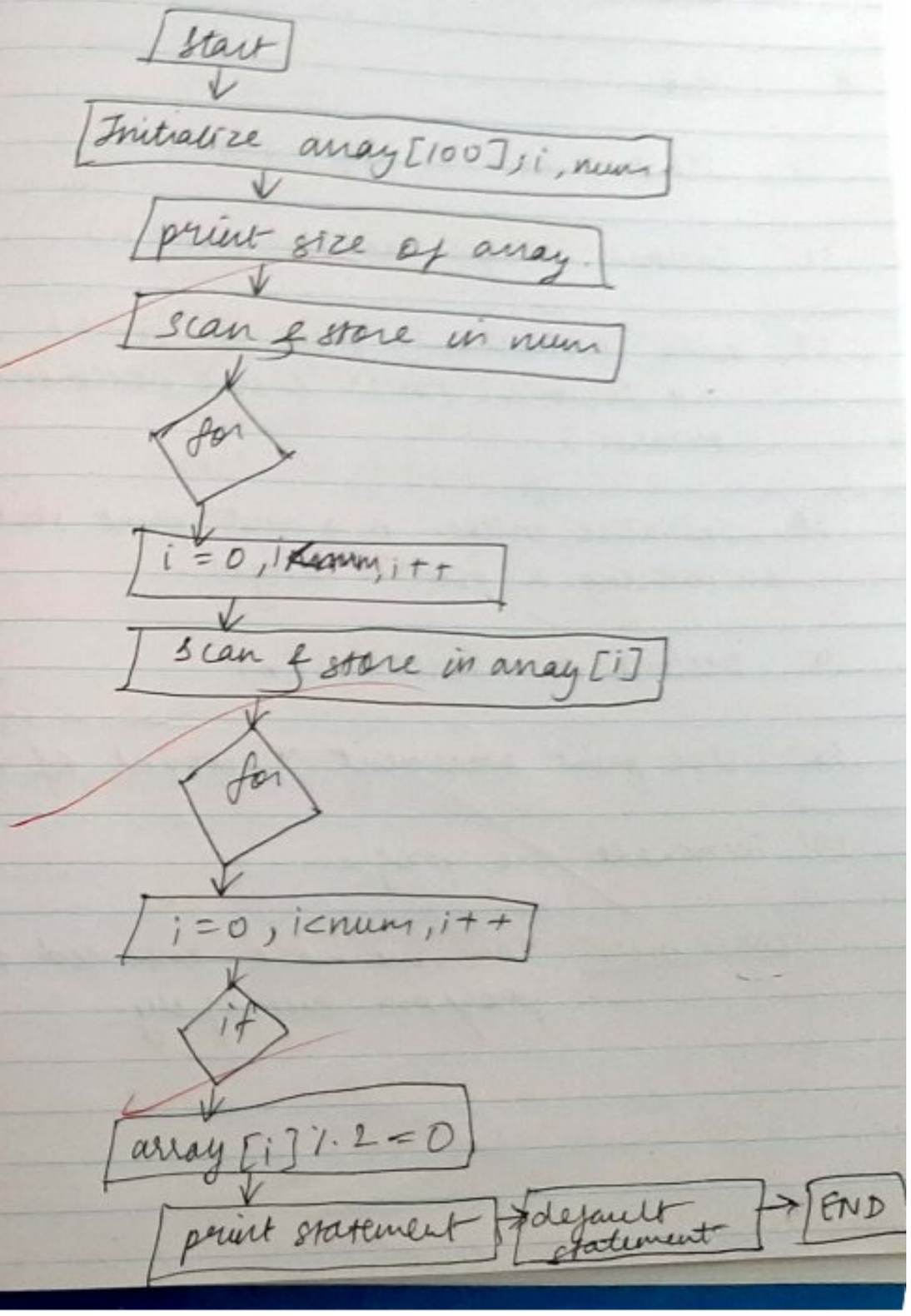
Output:

```
Enter the no. of elements: 4
Enter the no. again: 3
1. Enter no.: 4
2. Enter no.: 5
3. Enter no.: 6
Average = 5
sum is 15
```

* Flowchart :



* Flowchart for 5(b) :



Scanned with CamScanner

PRACTICAL-06

Aim:

(1) Factorial of a number using recursion.

Algo:

S1: Start Turbo C application.

S2: Declare integer factorial (int n).

S3: Using if statement, check if n>=1 & return n* factorial (n-1) & else statement, return 1.

S4: Initialize integer n, a and print statement by entering a positive integer.

S5: Declare a = factorial (n)

S6: Use print statement "factorial of %d is %d

S7: Terminate the program.

CONCLUSION: Thus, we have executed the program successfully.

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 n, a;
        clrscr();
        printf ("enter a positive integer : ");
        scanf ( "%d", &n);
        a = factorial (n);
        printf ("\n factorial of %d is : %d ", n,a);
        getch ();
}
```


output:

enter a positive integer : 6
Factorial of 6 is 720.

→ last page

Scanned with CamScanner

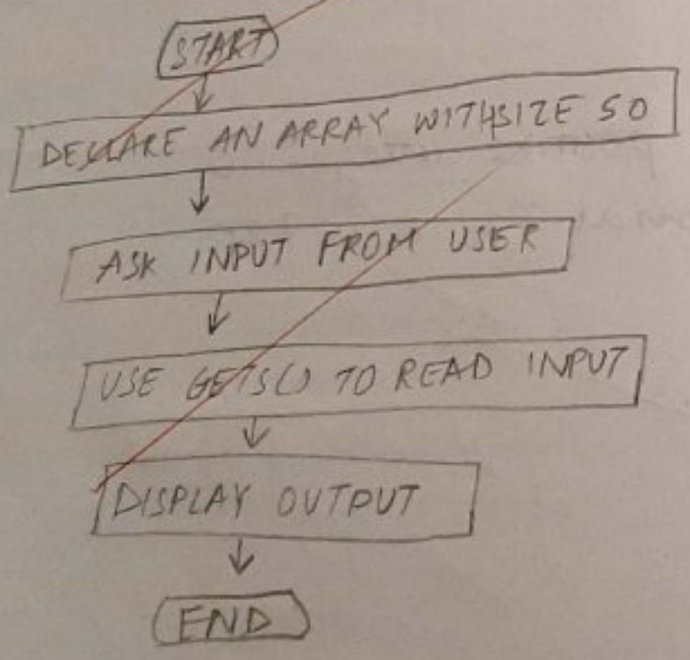
CODE:

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

output:

please enter your name: SakshiKulkarni
Sakshi. Kulkarni

* Flowchart:



Aim:
(2) WAP to which shows the use of gets() function

Algo:

S1: open Turbo C application.

S2: Initialize character name[50].

S3: Use print statement to display your name.

S4: Use gets(name) to display your name.

S5: Use print statement to finally display entered name on output screen.

S6: Terminate the program.

CONCLUSION: Thus, we have executed the program successfully.

Scanned with CamScanner

Aim:

(3) WAP to which shows the use of puts() function.

Algorithm:

S1: Open turbo c application.

S2: Initialize character name [60]

S3: Use print statement to enter your name.

S4: Declare gets(name).

S5: Use print statement to print the entered name.

S6: Use puts(name) to finally print enetered name onto the output screen.

CONCLUSION: Thus, we have successfully executed the program.

Khmodi

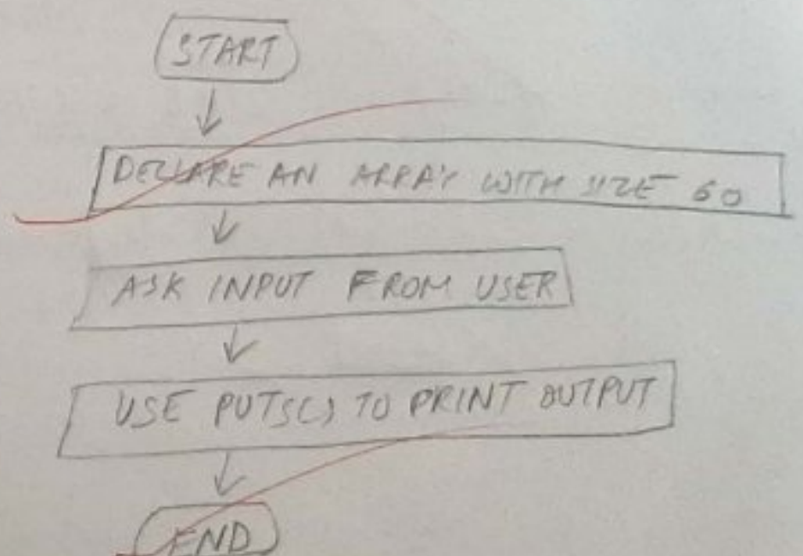
CODE:

```
#include < string.h >
#include < stdio.h >
#include < conio.h >
void main()
{
        char name [60];
        clrscr();
        printf("Enter your name:");
        gets(name);
        printf("\n Enter your name is ");
        puts(name);
        getch();
}
```

Output:

Enter your name: Sakshi
Sakshi

* FLOWchart:



Scanned with CamScanner

(A)

Aim: ~~Start turbo C~~ Swapping of array.

Algorithm:

S1: Start Turbo C program.

S2: Declare a function Prototype with 2 integer ptrs Pointers as argument before entering main()

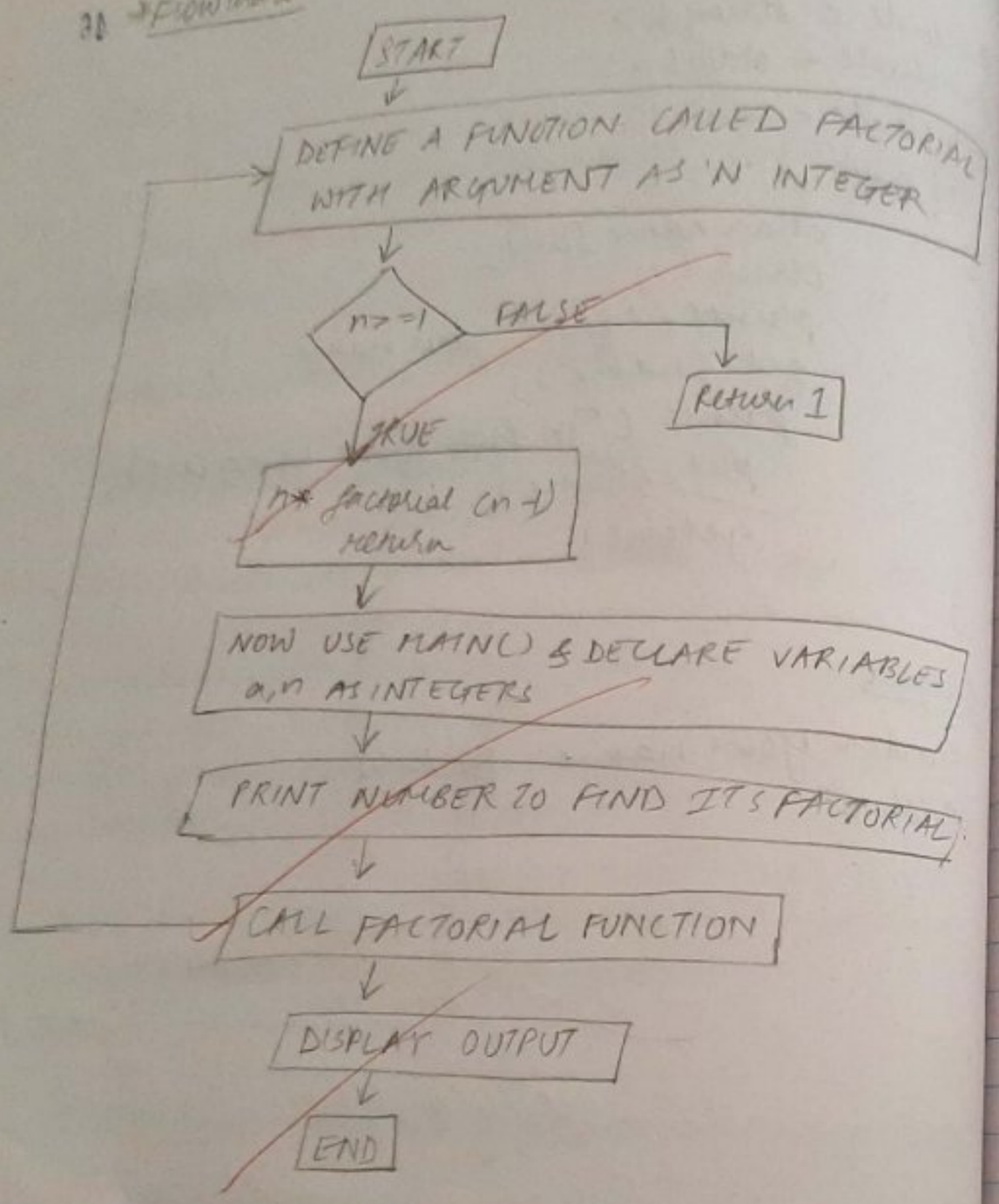
S3: Declare 2 variable and accept the value from user. Print repetative value using printf()

S4: For the address of the application as argument for the function.

S5: Print the respective value of variable.

S6: Use basic algorithm in the function defination but instead of normal variable user.

Flowchart



Scanned with CamScanner

Output:

Enter 2 nos. to be swapped: 29 99

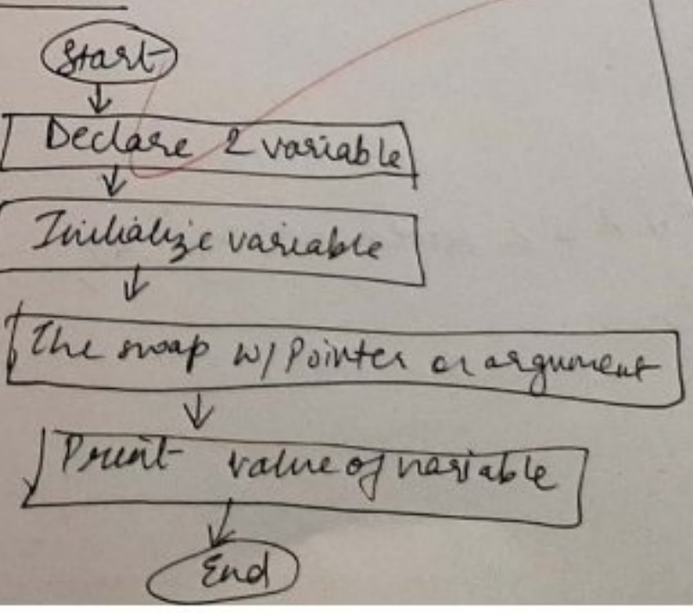
The no. before swapping are 29 and 99
The no. after swapping are 99 and 29.

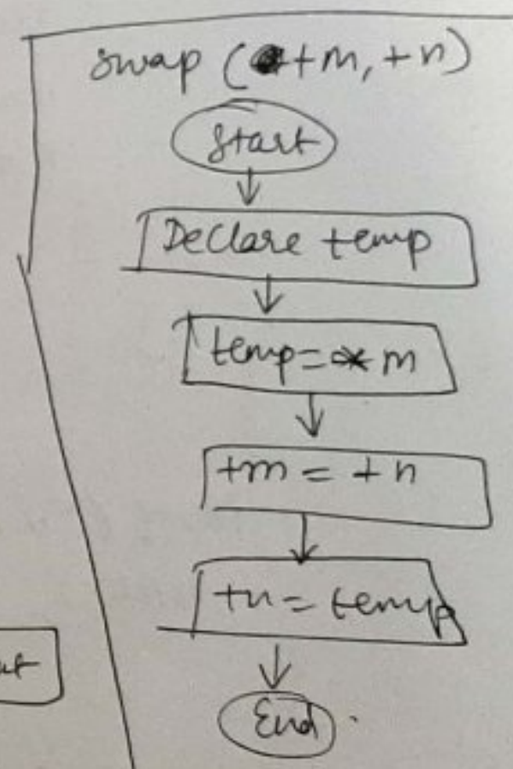
Conclusion: We have successfully executed the program

#code

```
#include <stdio.h>
#include <conio.h>
void swap (int *m, int*n);
void main ()
{
    int x,y;
    clrscr();
    printf ("Enter 2nos. to be swapped:");
    printf (" Value of before swapping are %d and
     %d respectively ",x,y);
    getch();
    }
    void swap (int*m,int*n)
    {
      temp = *m;
      *m = *n;
      *n=temp;
    }
```

· Flowchart:





Scanned with CamScanner

(B)

Aim: Sorting of array using Pointer.

Algorithm:

S1: Initialize an integer array and temp variable.

S2: Run nested loop or i=0 len(a) and OR i=0 to len(a)=i

S3: y+a >*a+1, swap two values using basic swapping logic.

S4: Print the swapped array.

S5: Terminate the program.

CODE:

```
void sort (int n, int*p)
#include<stdio.h>
#include<conio.h>
void main()
{
    int a[10]; i; temp;
    clrscr();
    for(i=0; i<10; i++)
    {
        for(j=0; j<10; j++)
        {
            if(*a > *a+1)
            {
                temp = *a+1;
                *a+1 = *a;
                *a = temp;
            }
        }
    }
Printf("%d is a sorted array", a);
  getch();
}
```

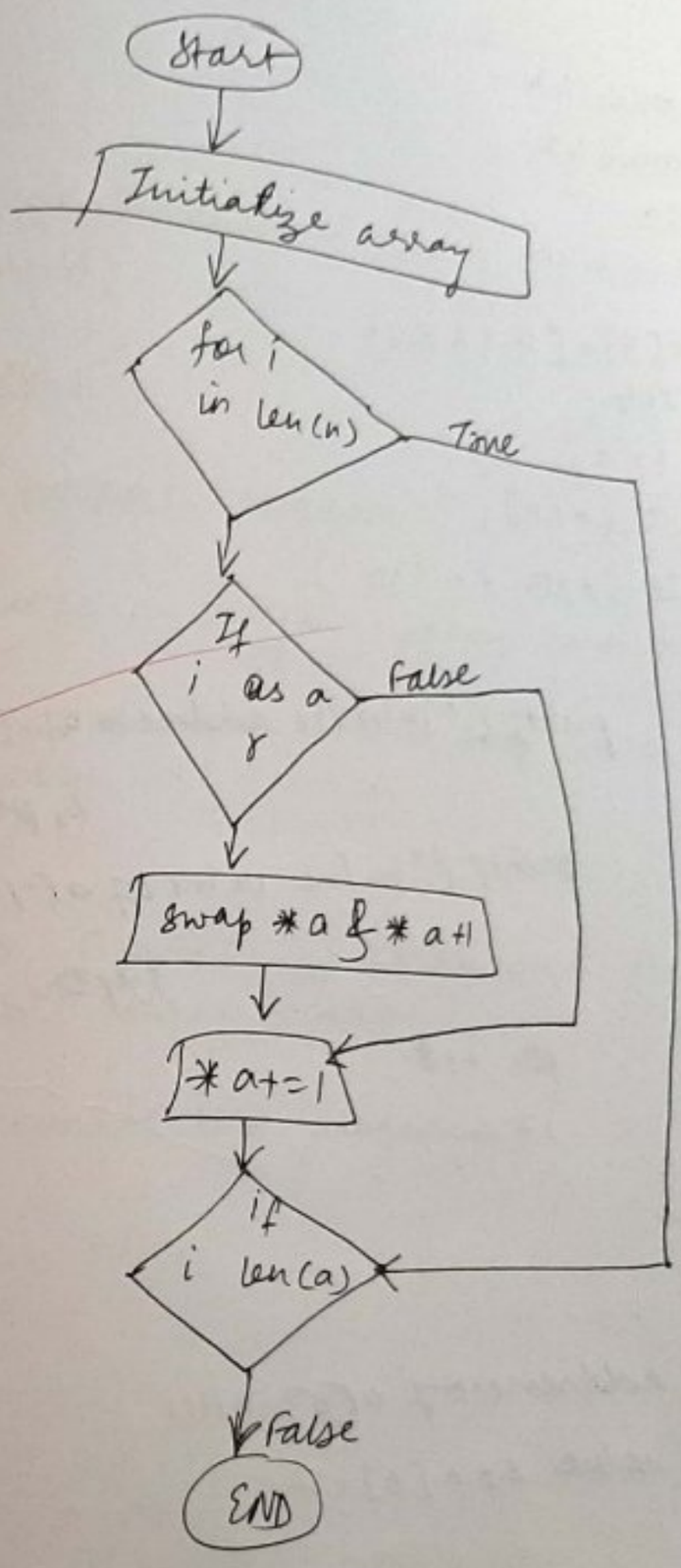
output:

Insert element into array:
1
9
5
8
2
3
6

11
{1, 2, 3, 5, 6, 8, 9, 11}
is the sorted array.

* conclusion: we have successfully executed program

#Flowchart:



Scanned with CamScanner

Code:

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

Output - The addresses of a[0] = 1111
The value of a[0] = 7

Aim: One-Dimensional array representation using Pointer.

Algorithm:

S1: open turbo C program.

S2: Initialize an integer array and a variable.

S3: Run a ~~variable~~ while loop with i=0 to length of array.

S4: Print the data of array and then use Pointer to print array location.

S5: Terminate the program.

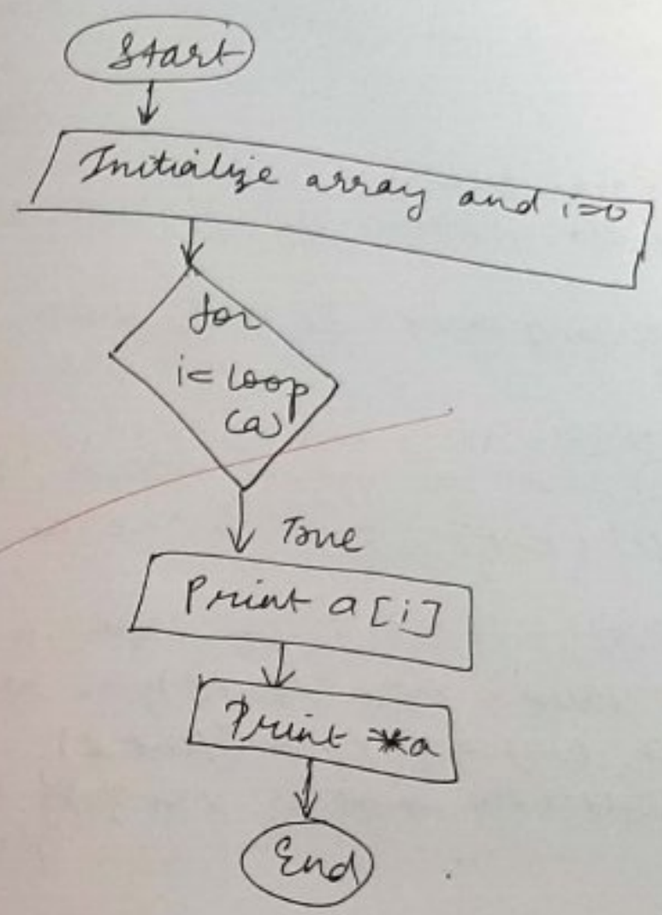
Scanned with CamScanner

* CONCLUSION:

we have succesfully executed the program.

Thanku

* Flowchart:



Scanned with CamScanner

(A) Program using string library functions for copying string.

Algorithm:

S1: Open the Turbo C applications.

S2: Input string from user and store it to some variable say text 1.

S3: Declare another variable to store copy of first string in text 2.

S4: Run a loop from 0 to end of string. The loop structure should be like: for(i=0, text1[i]!='\0'; i++)

S5: Inside the loop for each character in text 1 copy to text 2. Say: text2[i] = text1[i]

S6: finally after loop make sure the copied string end with NULL character ie; text2[i] = '\0'.

SE CODE:

```
#include<stdio.h>
#define MAX_SIZE 100

int main()
{
    char text1[MAX_SIZE];
    char text2[MAX_SIZE];
    int i;
    printf("Enter any string : ");
    gets(text1);
    for(i=0; text1[i]!='\0'; i++)
    {
        text2[i] = text1[i];
    }
    text2[i] = '\0';
    printf("First string = %s\n", text1);
    printf("First string copy = %s\n", text2);
    printf("Total characters copied = %d\n", i);
    return 0;
}
```

Output:

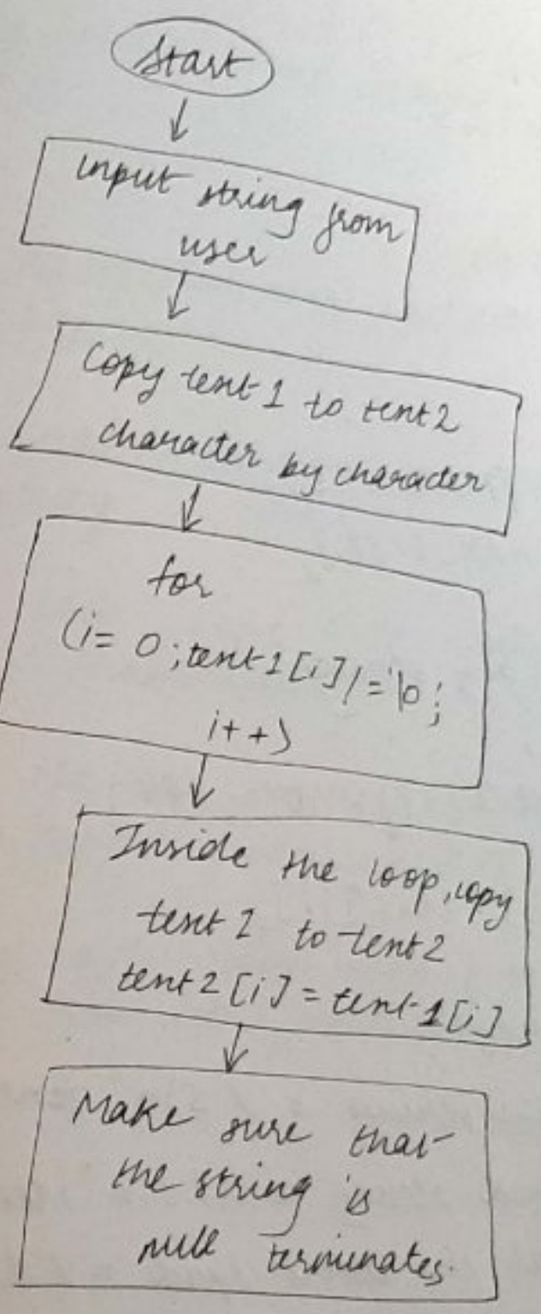
```
Enter any string : There are 7 days in a week
First string: There are 7 days in a week
String copy : There are 7 days in a week
Total characters copied: 26
```


Scanned with CamScanner

CONCLUSION:

Hence, we have successfully performed string copy program.

* Flowchart :-



Scanned with CamScanner

CODE:

```
#include<stdio.h>
#include<conio.h>
#include<stdlib
#define MAX_SIZE 100

int main()
{
    char text1[MAX_SIZE];
    char text2[MAX_SIZE];
    int i;
    printf("Enter any string");
    gets(text1);
    for(i=0; text1[i]!='\0'; i++)
    {
        text2[i] = text1[i];
    }
    text2[i] = '\0';
    printf("First string = %s\n", text1);
    printf("copied string is = %s\n", text2);
    printf("Total characters copied = %d\n", i);
    return 0;
}
```


(B) WAP to show string library functions, say strcpy().

Algorithm:

S1: Input string from user and store it to some variable say text1.

S2: Declare another variable to store copy of first string in text2.

S3: Run a loop from 0 to end of string. The loop structure should be like: for(i=0; text1[i]!=; i++);

S4: Inside the loop for each character in text1 copy to text2; say text2[i] = text1[i].

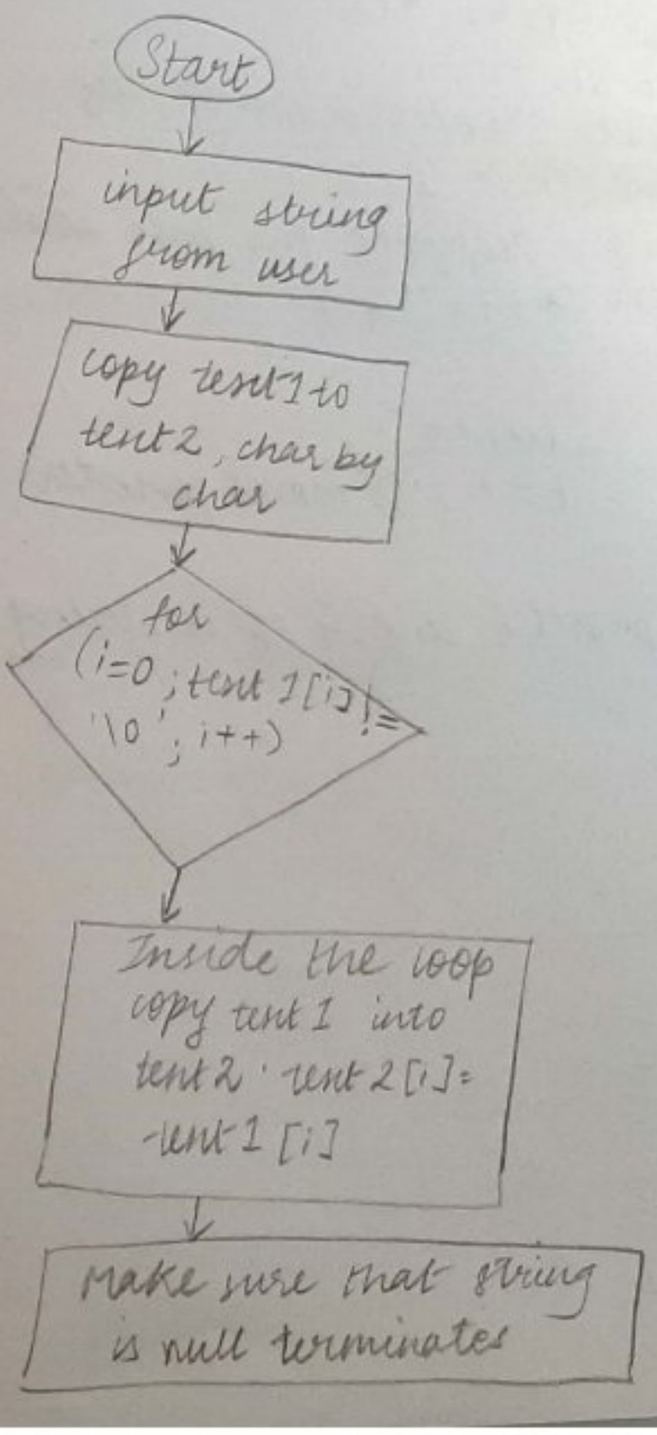
S5: Finally after loop make sure the copied string end with NULL character, i.e. text2[i]='\0'.

CONCLUSION: Hence, we have successfully executed the program.

output:

enter any string : C.S. is an interesting subject.
copied string is : C.S. is an interesting subject.
Total characters copied : 31.

flowchart:



Scanned with CamScanner

· CODE:

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


(c) WAP to show the length of a string without using string functions.

· Algorithm:

S1: * Input string from user say s. and

S2: input another string say *t from user.

S3: Assign length of initial string =0.

S4: Use print statement to enter a string.

S5: Show t=s to copy the base address of the string.

S6: Use while loop to check condition *t!='\0'; if true, increment string and length.

S7: If false, print length of string.

Scanned with CamScanner

- CONCLUSION: Hence, we have successfully executed the program.

Scanned with CamScanner

58

· CODE:

```
#include<stdio.h>
#include<conio.h>
struct student
{
        int id, CGPA;
        char name[25];
};
void main()
{
        struct student s[2];
        int i;
        clrscr();
        for(i=0; i<2; i++)
        {
            printf("Enter details of studends:");
            scanf("%d", s[i].id);
            scanf("%d", s[i].CGPA);
            scanf("%s", s[i].name);
        }
    for(i=0; i<2; i++)
    {
        printf("\n Entered details of students %d", i+1);
        printf("\n ID: %d \n CGPA: %d
                \n Name: %s", s[i].id, s[i].
                s[i].name);
    print("\n");
}
```

PRACTICAL-02

59

(A) Create a simple structure named student that holds the following variables: (i) ID, (ii) CGPA, (iii) Name

· Algorithm:

S1: Open the Turbo C application.

S2: Define a struct 'student' with id, CGPA and name variables.

S3: In the main() function, declare an array of student

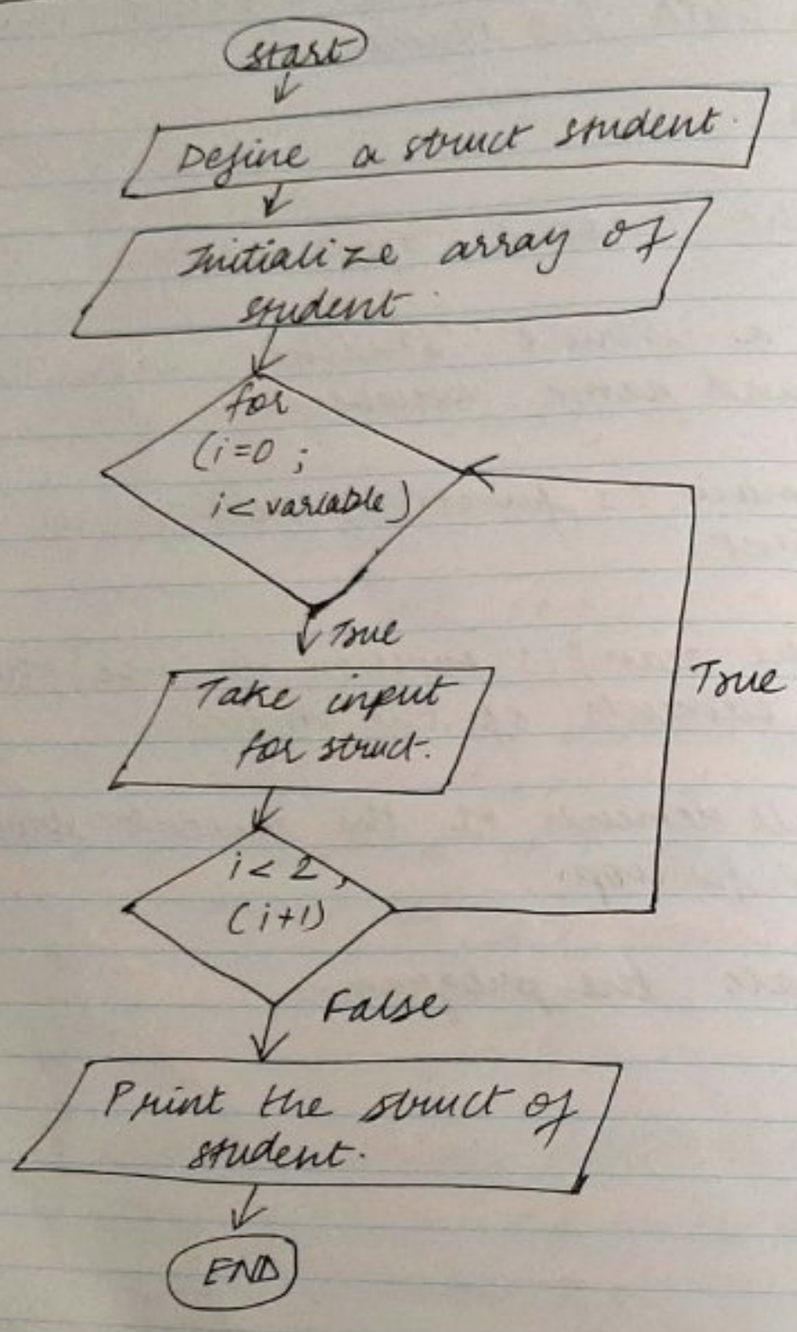
S4: Use the scanf() function to use the various ~~struct~~'s elements of the student.

S5: Print all elements of the student structure using a for loop.

S6: Terminate the program.

Scanned with CamScanner

Flowchart:



CONCLUSION: Hence, we have successfully executed the program.

```
    getch();
}
```

OUTPUT:

```
Enter details of student 1:
1896
5
Sakshi
Enter details of student 2:
1543
2
Nandini
Details of student 1:
ID : 1896
CGPA : 5
Name : Sakshi
Details of student 2:
ID : 1543
CGPA : 2
Name : Nandini
```


Scanned with CamScanner

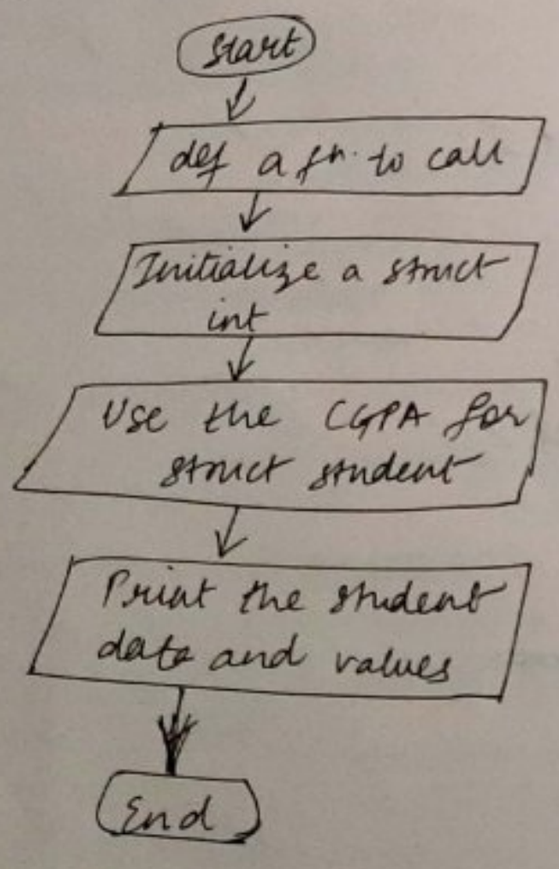
02 . CODE :

```
def calc(int n)
{
        return 60 * n
}
printf("Total marks obtained %d : calc(s[i]
```

. output :

Total marks obtained : 600
Total marks obtained : 540

. Flowchart :



61

(B) Use of struct with function ~~assertion~~ :

. Algorithm :

S1: Open Turbo C application.

S2: Define a struct and take the input as done previously.

S3: Now, define a function to calculate total marks obtained out of 600 using CGPA.

S4: Print the along with the student data.

S5: Terminate the program.

. CONCLUSION : Hence, we have successfully executed this program.

Scanned with CamScanner

(c) UNION:

Algorithm:

S1: Start Turbo C ~~prog~~ application.

S2: Use the union keyword to declare the union of various datatypes

S3: In the main body of the program, we use '.' operator to take input.

S4: Now, print all the data of the union.

S5: Terminate the program.

CODE:

```
#include <stdio.h>
#include <conio.h>
union student
{
    int roll no.
    char name[25], div, co[10];
    float perc;
}
void main();
{
    union student s;
    printf("Enter details of students %d", i+1);
    scanf("%d", s[i].id);
    scanf("%d", s[i].CGPA);
    scanf("%s", s[i].name);
}
for(i=0; i<2; i++)
{ printf("\n Entered details of student: %d", i+1);
  printf("\n ID: %d \n CGPA: %d \n Name: %s",
          s[i].id, s[i].CGPA, s[i].name);
}}
getch();
```


Scanned with CamScanner

23. Output:

Enter details of student 1:

1896
10
Sakshi

Enter details of student 2:

1544
9.5
Tanisha

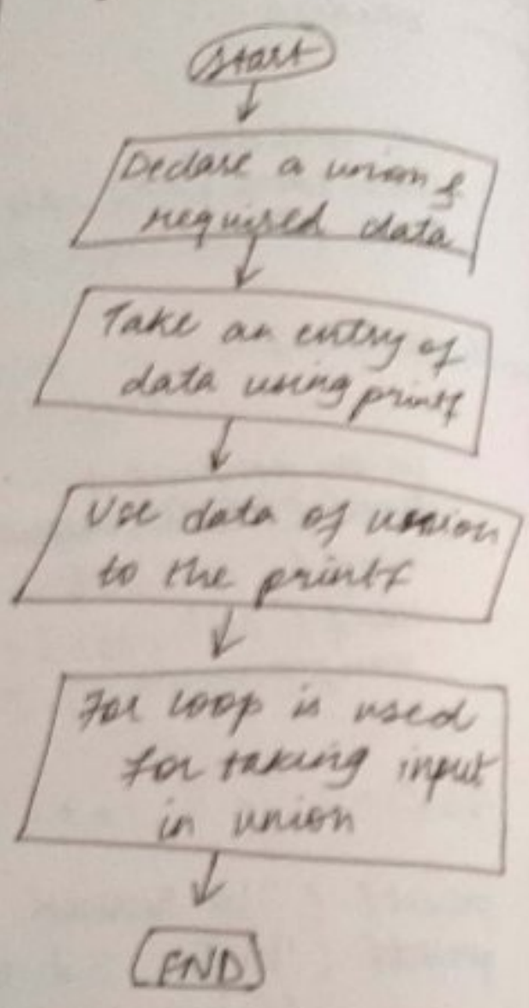
Details of student 1:

ID: 1896
CGPA: 10
Name: Sakshi

Details of student 2:

ID: 1544
CGPA: 9.5
Name: Tanisha

Flowchart



CONCLUSION: Hence, we have successfully executed the program using union.

Scanned with CamScanner