Code :

```
#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");
    scanf("%s", &mobile_no);
    printf("enter student's percentage:\n");
    scanf("%f", &percentage);
    printf("student's name:%s\n", name);
    printf("student's roll no:%d\n", roll_no);
    printf("student's mobile_no:%s\n", mobile_no);
    printf("student's percentage:%f\n", percentage);
    getch();
}
```


PRACTICAL : 1

Aim : Write a C program to understand the basic data-types.

Algorithm :

Step 1 : Declare a variable integer to store roll number, a float variable to store percentage and two string arrays for storing name and mobile number.

Step 2 : Print using the printf() method to ask for values and use scanf() to get the required values.

Step 3 : After receiving the values, print them one by one using the printf() statement.

Conclusion :

Then, the integer, character and float data types have been studied.

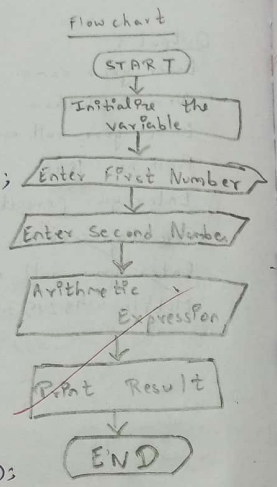
Xtramedi

Output :

```
Enter your name : Shubham
Name : Shubham
Enter your roll number : 1890
Roll no. : 1890
Enter your percentage : 80.20 %
Percentage : 80.20 %
Enter your roll number : 9821531021
Mobile : 9821531021
```

```c
#include <conio.h>
#include <stdio.h>
void main()
{ int a,b,result;
  clrscr();
  printf("Enter the two values");
  scanf("%d %d", &a, &b);
  result = a+b;
  printf("a+b= %d", result);
  result = a-b
  printf("a-b = %d", result);
  result = a/b;
  printf("a/b = %d", result);
  result = a* b;
  printf("\n a* b = %d", result);
  result = a % b;
  printf("\n a%b= %d", result);
  getch();
}
```

Flow chart



Output : Enter the two values : 15
10
a + b = 35
a - b = 5
a/b = 1
a*b = 150
a %b = 5

* Aim : Write a C program on operators and Expression.

(i) Expressions

Algorithm :

Step 1 : Declare three variables, two to accept the values and one to store result.

Step 2 : Use the scanf() function to accept two values from the user.

Step 3 : Use the values operators such as '+', '-', etc. to perform arithmetic operator on the variables and store the value in the third variable and print it.

* Conclusion :

The various expressions and binary operators have been studied.

Viva

(ii) Ternary Operator :

* Algorithm :

Step 1 : Declare 3 integer variables.

Step 2 : Use the scanf() method to accept the values.

Step 3 : Use the ternary operator to store the value of the greater number in the third variable.

Step 4 : Print the greater number.

* Conclusion :

The ternary Operator works like a conditional statement 'if-else' and stores the resultant value in a derived variable.

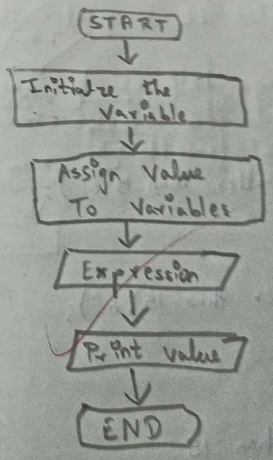
Approved

```
#include <conio.h>
#include <stdio.h>
void main()
{ int a,b,x;
  printf("Enter any two numbers");
  scanf("%d%d",&a,&b);
  x = a>b ? a : b;
  printf("The greater number is",a);
  getch();
}
```


Output :

```
Enter any two numbers : 15
                        20
The greater number is 20
```

Flow Chart :



PRACTICAL 3

Aim: Write a C program 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 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 and if condition is false skip the if statement and print I am not in if.

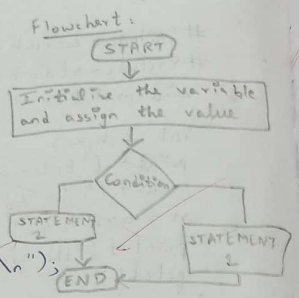
b] Write a program in C to explain if else statement

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 le than 15 or if statement is false than prin 20 is greater than 15.

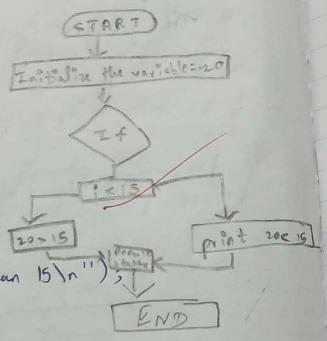
Flowchart:



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

Output: 20 is greater than 15.

PS

c] Write a program in C to explain nested-if statement.

Algorithm:

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

Step 2: Now use nested-if logic to compare of given number is greater or not.

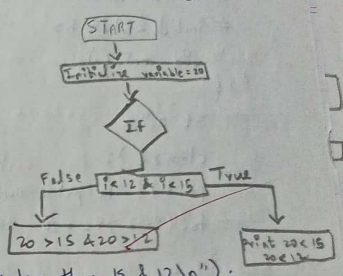
Step 3: If condition is true then go to second condition of second condition is also true then print that 20 is greater than 15 & 12. If one of conditions are not true then skip the part and print 20 is greater than 15 & 12.

* Conclusions: These programs help us to understand the workings of if, if-else and nested-if conditional statements.

Rhmadi

c] Code: Nested if

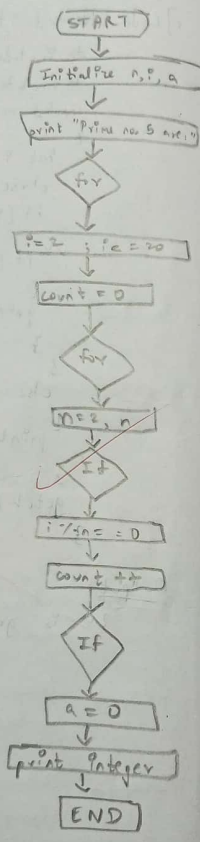
```
#include <stdio.h>
#include <conio.h>
void main()
{
  int i = 20;
  clrscr();
  if (i < 15)
  {
    if (i < 12)
    {
      printf("20 is less than 15 & 12\n");
    }
  }
  else
  {
    printf("20 is greater than 15 & 12\n");
  }
  getch();
}
```



Output:

20 is greater than 10 & 12

```
#include <conio.h>
#include <stdio.h>
void main()
{
int n,i,a
clrscr();
printf("The prime number 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 nos are:- 2,3,5,7,11,13,17,19

PRACTICAL NO. 4

Aim: To display the prime numbers using for loop

Algorithm:

Step 1: Initialize the variable out of which two are loop variable and one are count variable.

Step 2: Initialize a for loop. I to 50 let the count variable to be zero.

Step 3: Next another loop within the loop in step 2 that goes to 2 to the first variable % 2

Step 4: Use the if condition statement to check whether (first loop variable % 2nd loop variable 10, if true increment count variable by 1

Step 5: Come out of the second loop and check whether the count variable is 0 of true print the number.

Step 6: Terminate the program.

conclusion: Thus we have successfully executed prime numbers using for loop

(ii) Write a C program on Fibonacci series.

Algorithm:

Step 1: Start Turbo C

Step 2: Declare the variables n1, n2, n3, i, number.

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

Step 4: Enter the no. of terms of fibonacci series to be printed.

Step 5: Print first 2 ~~numbers~~ terms 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

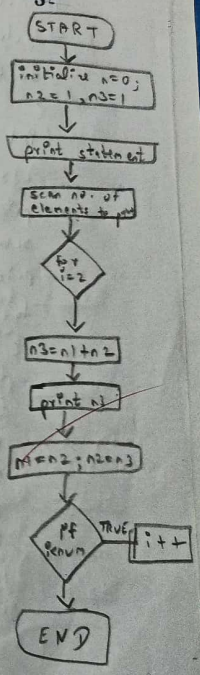
Conclusion: Thus, we have successfully executed fibonacci series on Turbo C.

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

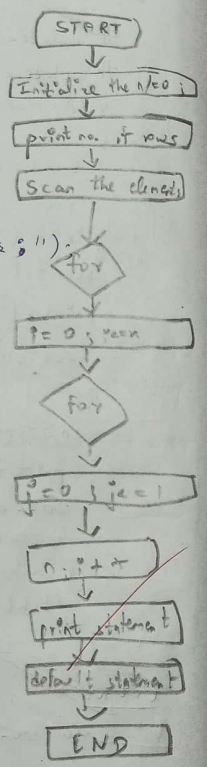
Output:

Enter number of elements 10

0 1 1 2 3 5 8 13 21 34 55



```c
#include <stdio.h>
#include<conio.h>
  void main()
  {
   int n1=0,r,i,j;
   clrscr();
   printf("Enter the number of rows:");
   scanf("%d",&r);
   printf("\n");
   for(i=0;i<=r;i++)
   {
    for(j=0;j<=i;j++)
    {
    n1++;
    printf("%d",n1);
    }
    printf("\n");
   }
   getch();
  }
```



Output:

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


Write the C program on following expression.

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

Step 1: Start the turbo C program.

Step 2: Declare the variable rows, i, j, number = 1

Step 3: Display the number of rows.

Step 4: Enter the for loop i=1; i<=rows; i++.

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

Step 6: Display the no. as per user enter the sequence from i=1

Step 7: Increment number from 1

Step 8: Display the space

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

Nanaadi

PRACTICAL : 5

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

Algorithm :

Step 1 : Start Turbo C application.

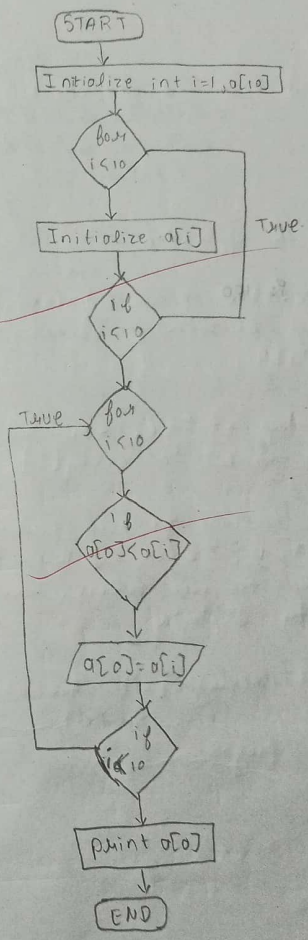
Step 2 : Declare the variable i and integer array a [10]

Step 3 : Enter the for loop at i = 0, i < 10 and 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 statement 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.



Output:

```
Enter the elements:
12
23
2
12
2
55
3
22
100
The largest number is 100
```

Source code:

```
#include <stdio.h>
#include <conio.h>
  void main()
{
 int a[10], i;
 clrscr();
 printf("Enter the elements of list: \n");
 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()
}
```

C program to find even and odd numbers using array

Algorithm :

Step 1 : Start Turbo C application

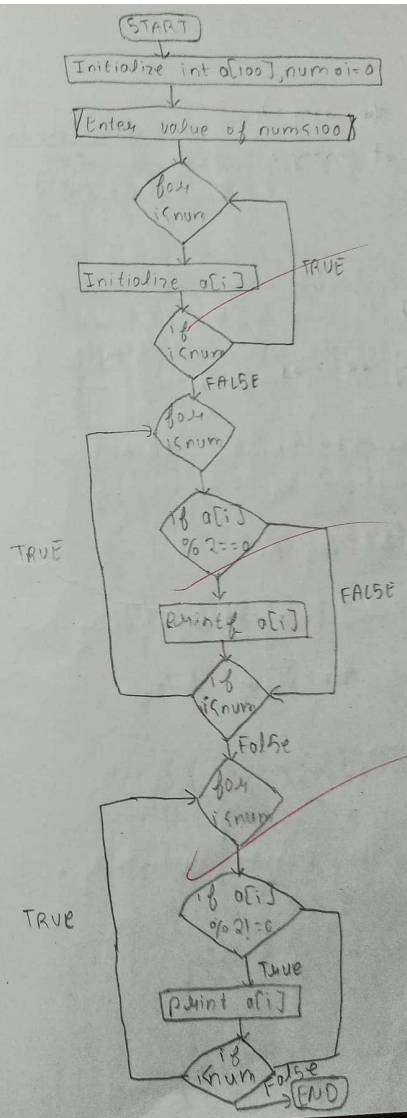
Step 2 : Declare the variable i, num and array[100], print the message for entering element, store it in num variable.

Step 3 : Enter the for loop at i=0, i<num and increment it. store the element in array[i].

Step 4 : Give the print message for even numbers. Use for loop at i=0; i<num and increment it. Use if condition for even numbers. Give the print message for displaying.

Step 5 : Give the print message. Exit for loop

Step 6 : Terminate the program.



Output:

```
Enter the size of the array : 4
Enter the element of array :
10
20
30
35
Even no. of array
10    20    30
odd no. of array are : 35
```

Source code:

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

Aim: C program to find average & sum using array.

Algorithm:

Step1: Start Turbo C application

Step2: Declare the float, int variable n, i. initialize num[100], sum=0.0, avg.

Step3: Avg using for loop at i=0; i<n; i++. Give print message and increment i by 1.

Step5: Average is sum divided by n.

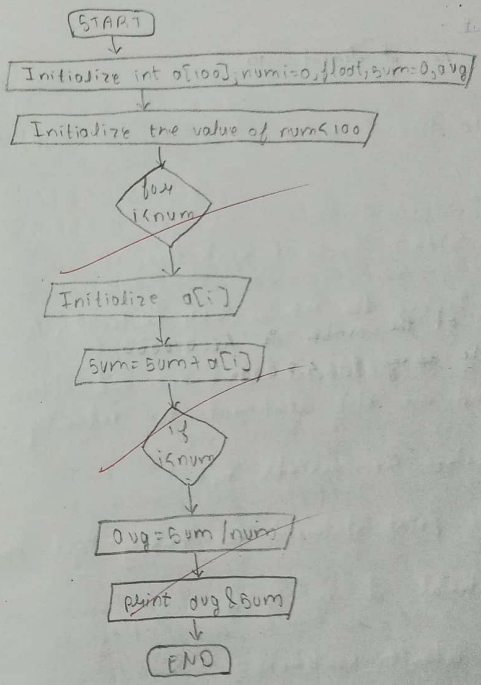
Step4: Declare sum variable and store it by adding num[i].

Step6: Give print statement for average & sum.

Step7: Terminate the program.

Conclusion:
Thus, we have executed the program successfully.

Ahmed



Output :

```
Enter no. of elements : 10
2
3
6
5
10
1
2
1
7
11
The sum of the number is 48.000000
and the average is 4.800000
```


source 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++);
scanf("%d", &num[i]);
sum = sum+sum[i];
}
avg = sum/n;
printf("Average = %f", avg);
printf("Sum = %2f", sum);
getch();
}
```

pb

PRACTICAL 6:

Aim: To find factorial of a number using recursion.

Algorithm:

Step 1: START Turbo C application.

Step 2: Declare the int variable factorial, n

Step 3: Use if conditional statement and return fatorial, and use else statement for returning.

Step 4: Declare int variable n, a.

Step 5: Use print statement for taking input from user.

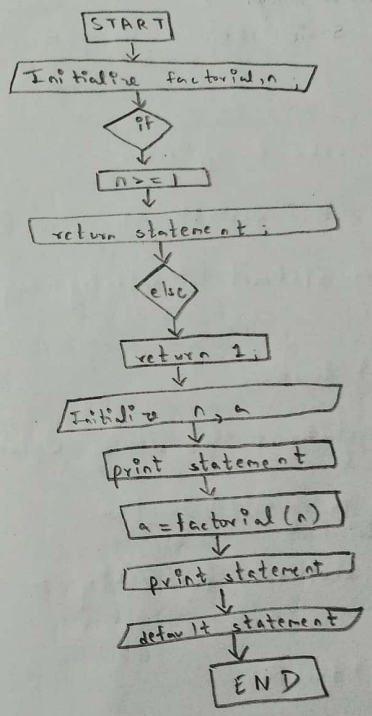
Step 6: Factorial of n is a

Step 7: Use default statement.

Step 8: Display the output.

Step 9: Terminate the program.

Flowchart:



PRACTICAL 6:

Aim: To find factorial of a number using recursion

Algorithm:

Step 1: START Turbo C application.

Step 2: Declare the int variable factorial, n

Step 3: Use if conditional statement and return factorial and use else statement for returning.

Step 4: Declare int variable n, a.

Step 5: Use print statement for taking input from user

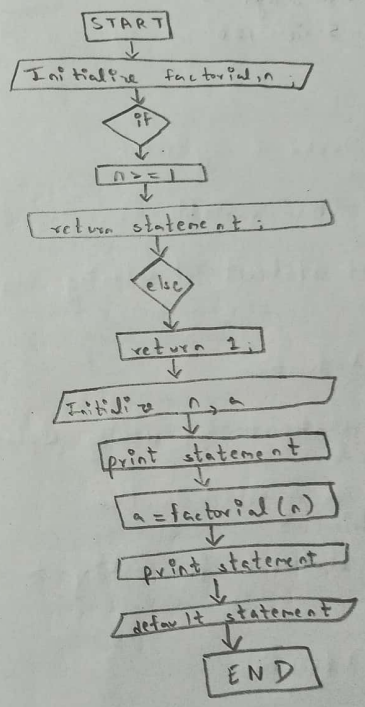
Step 6: Factorial of n is a

Step 7: Use default statement.

Step 8: Display the output.

Step 9: Terminate the program.

Flowchart:



Output:

```
Enter a positive integer : 5
Factorial of 5 is 120
```

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

(B) Aim : C program which shows the use of get th

Algorithm :

Step 1 : Start Turbo C application.

Step 2 : Declare char variable ch.

Step 3 : Use print statement of entering any key to continue.

Step 4 : Use getch() function.

Step 5 : Use print statement for entering alphabet.

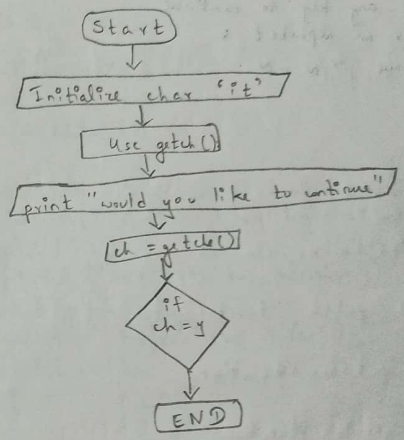
Step 6 : Assign ch to getche() function.

Step 7 : Use print statement for asking to continue the process.

Step 8 : Use getchar() function.

Step 9 : Terminate the program.

Flowchart :



Output :

```
Press any key to continue
Enter an alphabet s
continue Y/N -N
```

Source code :

```
#include<stdio.h>
#include<conio.h>
void main()
{
char ch;
printf("\n Press any key to continue");
getch();
printf("\n Enter an alphabet");
ch = getche();
printf("\n continue Y/N");
getch ar();
}
```

(c) Aim: C program for showing the use of put function.

Algorithm:

Step 1: Start Turbo C application

Step 2: Initialize char ch as b

Step 3: Use putch function for ch.

Step 4: Use putchar function.

step 5: Use getchar function.

Step 6: Terminate the program.

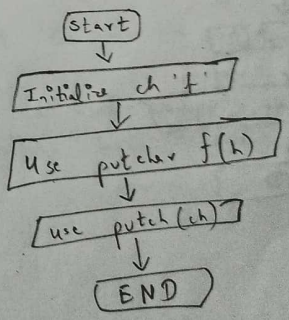
Source Code:

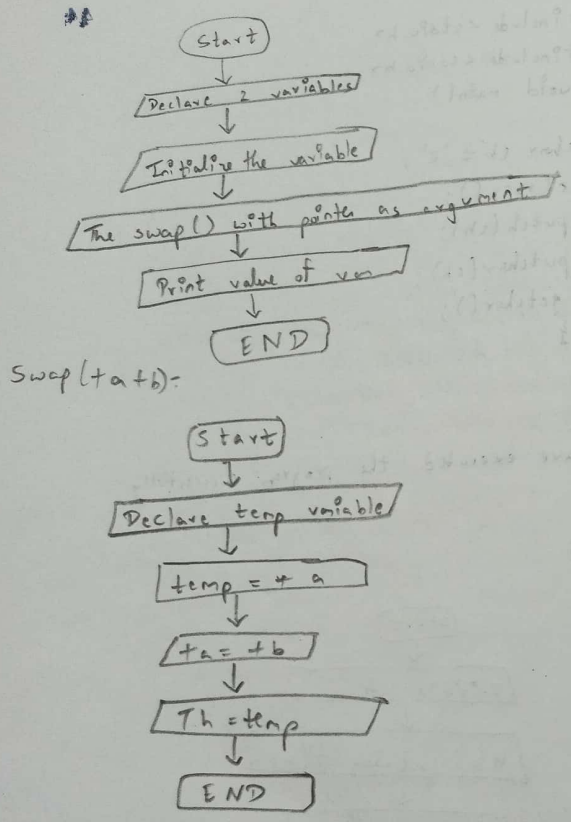
```
#include <stdio.h>
#include <conio.h>
 void main()
{
 char ch = 'b';
 clrscr();
 putch(ch);
 putchar(ch);
 getchar();
}
```

Conclusion:

Thus, we have executed the program successfully

Flowchart:





Swap(*a,*b):-

PRACTICAL : 7

Aim : Write a program to find the swapping of 2 Nos (Pointers)

Algorithm :

Step 1 : Start the turbo C 7 application.

Step 2 : Declare a function prototype with two integer pointer as argument before entering main().

Step 3 : Declare 2 variable and accept thus value from the user, print the reputive value using printf().

Step 4 : Pair the address of the variable as argument for the function.

step 5 : Print the repetive value of the variable.

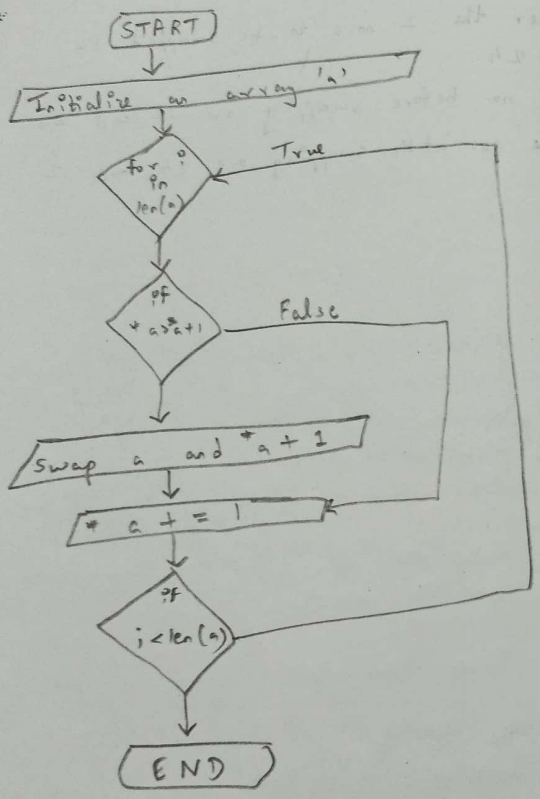
Step 6 : Use the basic swapping algorithm in the partion definition but instead of normal valuable use.

Source code:

```
#include <conio.h>
#include <stdio.h>
void swap (int* m, int *n);
void main()
{
   int x, y;
   clrscr()
   printf("Enter the two number to be swapped:");
   scanf("%d %d", &n, &m);
   printf("The value before swapping are %d and
          %d respectively x, y");
   getch();
}
   void main(int* m, int* n)
{
   temp = *m;
   *m = *n;
   *n = temp;
}
```

Output:


```
Enter the 2 no.s to be swapped: 12
                                24
The no. before swapping are 12 and 24
The no. after swapping are 24 and 12.
```



(i) (ii) Sorting of arrays using pointer

Algorithm :

Step 1 : Initialize an integer array, i, j and temp variable.

Step 2 : Run a nested loop of i = 0 to len(a) and of j = 0 to len(a) - i.

Step 3 : if *a > *a + 1, swap the two values using basic swapping logic.

Step 4 : Print the swapped array.

Step 5 : Terminate the program.

Code :

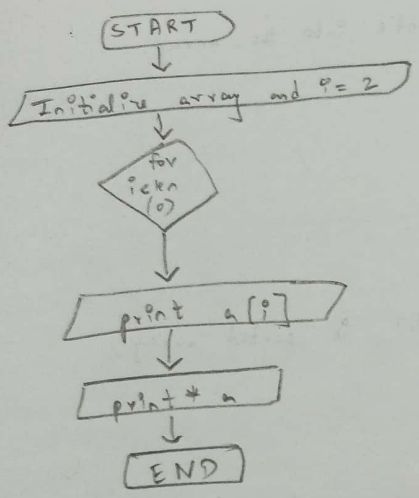
```
#include <stdio.h>
#include <conio.h>
void main()
{
  int a[10]; i,j, temp;
  clrscr();
  for(i=0; i<10; i++)
  {
    for(j=0; j<10-i; j++)
    {
      if(*a > *a+1)
      {
        temp = *a+1;
        *a+1 = *a;
        *a = temp;

      }
    }
  }
  printf("%d is the sorted array",a);
  getch()
}
```

Output :

Insert elements into the array.

2
5
9
8
6
4
3
1

1,2,3,4,5, 6,8,9 is sorted array.



(iii) Write a program to find one dimensional array using pointer.

Algorithm :

Step 1 : Start the turbo C 7 application.

Step 2 : Initialize an integer array and a variable.

Step 3 : Run a for loop with a = 0 to length of array.

Step 4 : Print the data of the array and then use pointer to print the memory location.

Step 5 : Terminate the program.

Code :

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

Conclusion: The program to find one-dimensional array using pointer is done & subsequently.

Output :

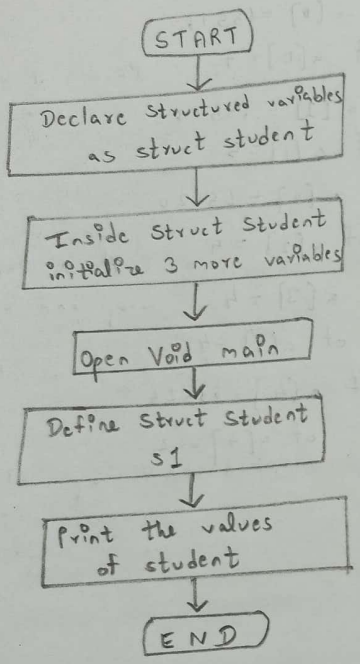
The address of a[0] = 65516
The value of a[0] = 7

The address of a[1] = 65518
The value of a[1] = 5

The address of a[2] = 65520
The value of a[2] = 9

The address of a[3] = 4
The value of a[3] = 4

The address of a[4] = 65524
The value of a[4] = 2



PRACTICAL - 8

A] Aim : Create a simple structure named on student that holds following variable : id, CGPA, Name.

Algorithm :

Step 1 : Start Turbo C

Step 2 : Declare the structured variable as 'Struct-Student'.

Step 3 : Initialize the struct student with 3 more variables inside the as 'int id', 'float CGPA', 'char name[10]'.

Step 4 : Now, inside void main() define struct student s1;

Step 5 : Print the details of the student such as id, CGPA, Name.

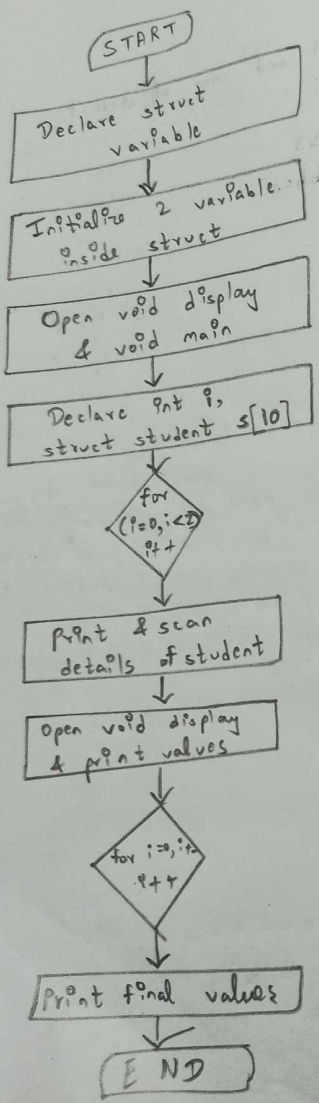
Step 6 : Terminate the program.

Source code:

```
struct student
{
  int id;
  float CGPA;
  char name[10];

};
void main()
{
  struct student S1;
  printf("Enter id, CGPA and name of student");
  scanf("%d %f %s", &S1.id, &S1.CGPA, S1.name);
  printf("\n id = %d", S1.id);
  printf("\n CGPA = %f", S1.CGPA);
  printf("\n Name = %s", S1.name);
}
```


Output:

```
Enter id, CGPA and name of student
  id = 1
CGPA = 8.333
 Name = Shubham
```



B] W.A.C.P which will demonstrate use of structure and function.

Algorithm:

Step 1: Start the Turbo C Application.

Step 2: Initialize the struct student with two or more variables (int roll and char name[10])

Step 2: Declare the structured variable as struct student.

Step 4: Now inside void main display and void main declare int i, struct student s[10]

Step 5: Use the for loop fy. entering details of students upto 2 students and not more than that.

Step 6: Print the details of the student.

Step 7: Open void display again and print the values using condition & printf.

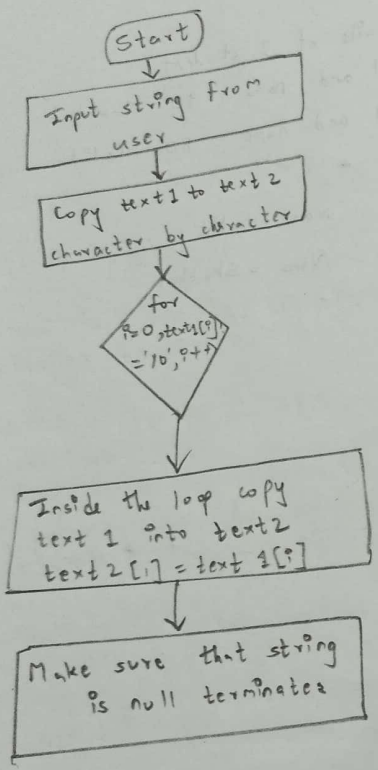
Step 8: Terminate the program.

Code :

```
#include <stdio.h>
 struct student
{
 int roll;
 char name[10];
};
 void display(struct student s1[10]);
 void main()
{
 int i;
 struct student s[10];
 clrscr();
 printf("\n Enter details of 2 students");
 for(i=0; i<2; i++)
 {
 print f("\n Enter roll and name");
 scan f("%d %s", &s[i] roll, s[i]name);
 }
 display(s);
 getch();
}
void display(struct student s[10])
{
 int i;
 printf("\n ********** \n");
 for(i=0; i<2; ;++)
 printf("\n roll=%d \tName=%s", s1[i].
                roll, s1[i].name);
```

Output :

```
Enter details of 2 student
Enter roll and name : 90  Shubham
Enter roll and name : 80  Lalit
* * * * * *
Roll = 80     Name = Lalit
Roll = 90     Name = Shubham
```



PRACTICAL : 9

[A] Aim : WAP to copy one string into another string

Algorithm :

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

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

Step 3 : Run a loop from 0 to end of string. The loop structure should be like
for (i = 0; text1[i] != '\0'; i++)

Step 4 : Inside the loop for each character in text 1 copy to text 2. say text2[i] = text1[i]

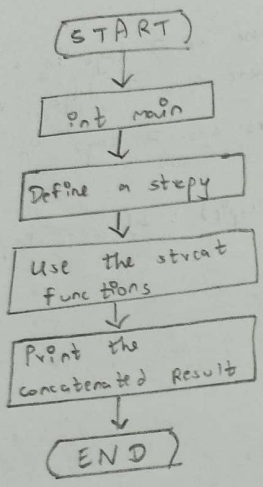
Step 5 : Finally after loop make sure the copied string ends with null character i.e. text2[i] = '\0';

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



(13) Write a program which will demonstrate the use of string library function.

strcat : The strcat() function will append a copy of the source string to the end of destination string. The strcat function takes 2 argument : 1) dest
2) src

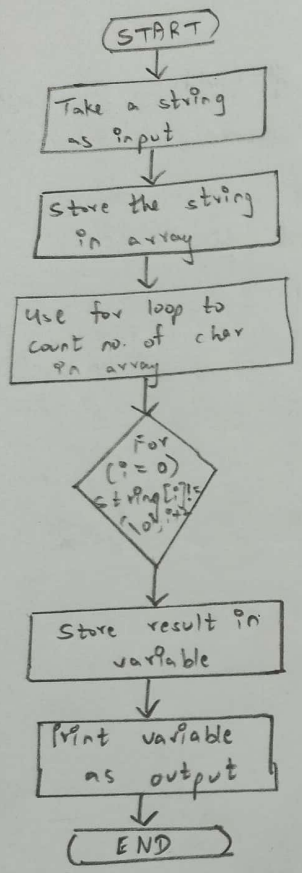
The strcat function returns a pointer (where the resulting concatenated string resides).

Code :

```
#include <stdio.h>
#include <string.h>

int main (int argc, const char * argv[])
{
  char example[100];
  strcpy (example, "Rahul");

  };
  strcat (example, " is over 18");
  strcat (example, " years old");

  printf ("%s\n", example);
  return 0;

}
```

Output :

Rahul is over 18 years old.



c] WAP which displays the length of a string without using string function.

Algorithm:

Step 1: Take a string as input and store it in the array.

Step 2: Using for loop count the number of characters in array and store the result in the variable.

Step 3: Print the variable as output.

#2

Code:

```
#include <stdio.h>
void main()
{
    char string[50];
    int i, length = 0;

    printf("Enter a string\n");
    gets(string);
    for i=0; string[i]!='\0'; i++)
    {
      length ++;
    }

printf("The len of str is the no. of
                        characters in:")
printf("so the len of: %s=%d\n", string,
                        length).
```

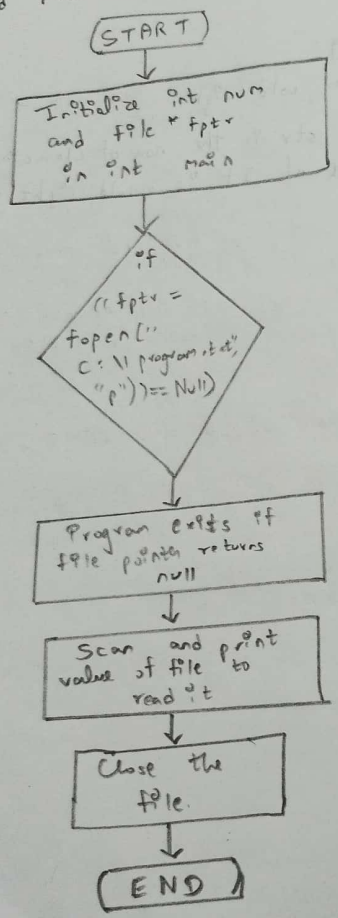
Output:

Enter a string:

It is a cold night

The length of str is the no. of characters in it
So the length of. It is a cold night = 18.

Read from text file and close it



PRACTICAL 10

Aim: Program for file open, file read and file close.

fopen() → Opens a existing file or create a new file for use.

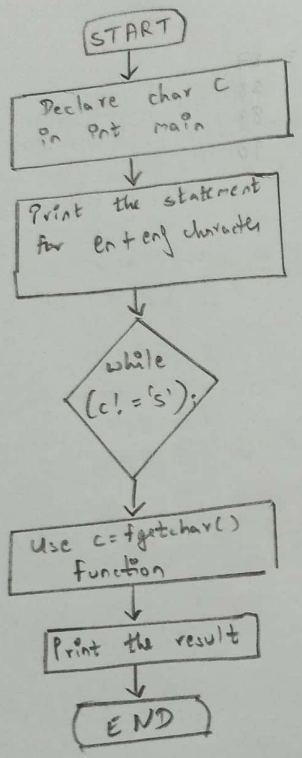
fread() → Reads a Record from a file.
fclose() → Closes a file.

Code for reading from a text file / opening / closing
Text file is (.txt and its contents are

```
#include <stdio.h>
#include <stdlib.h>
  int  main()
{
   int num;
   FILE * fptr;
if ((fptr = fopen("C:\\program.txt","r"))==Null
 {
  print f("Error ! opening file");
   exit(1);
}
  fscan f("fptr,"%d",&num);

  print f("value are = %d",num);
   fclose (fptr);
   return 0;
}
```

Output:

Values are = 87
88
89
90



B] Aim: WAP for fget c(), fgetchar(), fgetchar() function

Algorithm:

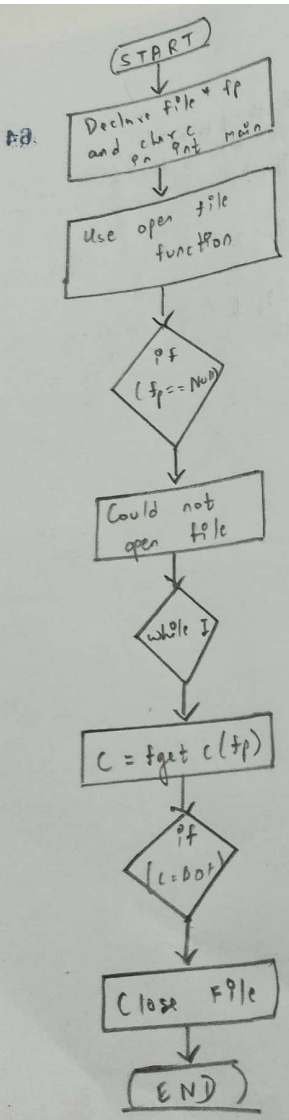
- Fgetchar is a file handling function.
- It is used to read a single character from keyboard input.

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 $
```



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

Code :

```
# include <stdio.h>
int main()
{
file *fp;
char c;
printf("Opening file test.c gn read mode");
fp = fopen ("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);
}
```

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

Output:

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