MRCET CAMPUS

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS INSTITUTION - UGC, GOVT. OF INDIA)

Affiliated to JNTUH; Approved by AICTE, NBA-Tier 1 & NAAC with A-GRADE | ISO 9001:2015

Maisammaguda, Dhulapally, Komaplly, Secunderabad - 500100, Telangana State, India

LABORATORY MANUAL & RECORD

Name:..

Roll No:................Branch:...

Year:..................Sem:...

MRCET CAMPUS

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS INSTITUTION - UGC, GOVT. OF INDIA)

Affiliated to JNTUH; Approved by AICTE, NBA-Tier 1 & NAAC with A-GRADE | ISO 9001:2015

Maisammaguda, Dhulapally, Komaplly, Secunderabad - 500100, Telangana State, India

Certificate

Certified that this is the Bonafide Record of the Work Done by

Mr./Ms..Roll.No.................of

B.Tech I year .... Semester for Academic year 2021 - 2022

in..Laboratory.

Date: Faculty Incharge HOD

Internal Examiner External Examiner

INDEX

S.No	Date	Name of the Activity/Experiment	Grade/ Marks	Faculty Signature

Preface

- C builds a strong foundation for programming. Helps to understand the fundamentals of Computer Theories. The goal of programming is to come up with software that will be able to do some tasks.
- This manual was developed specifically for freshmen students taking up their first course in programming. Its aim is to supplement classroom lectures by focusing on C programming. Topics are arranged based on the order of class room discussion.
- It is assumed that the student will be working under the Linux environment and programming using GNU C or the Borland Turbo C/C++ compiler under the Windows environment. Coding standards are to be followed.
- Case Studies are also included at the end of the manual which will help students to implement the concepts learned by them .They can also develop a mini project based on their knowledge and understanding.
- C programming is a basic programming language which should be learned by every student of engineering. Students will have a working knowledge on basic data structures through c programming.

PROGRAM OUTCOMES

A B.Tech –graduate should possess the following program outcomes.

1 **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.

2 **Problem analysis**: Identify ,formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3 **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4 **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5 **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6 **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7 **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8 **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9 **Individual and teamwork**: Function effectively as an individual, and as a member or leader in diverse teams ,and in multi disciplinary settings.

10 **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large ,such as, being able to comprehend and write effective reports and design documentation ,make effective presentations ,and giveand receive clear instructions.

11 **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multi disciplinary environments.

12 **Lifelong learning**: Recognize the need for and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY

I Year B. TECH - I- SEM **L/T/P/C**

-/-/3/1.5

(R20A0581) PROGRAMMING FOR PROBLEM SOLVING LAB

Course Objectives:

1. To understand the various steps in Program development.
2. To understand the basic concepts in C Programming Language.
3. To learn how to write modular and readable C Programs.
4. To learn to write programs (using structured programming approach) in C to solve problems.
5. To introduce basic data structures such as lists, stacks and queues.

Week 1:

a. Write a program to find sum and average of three numbers
b. Write a program to calculate simple interest(SI) for a given principal (P), time (T), and rate of interest (R) (SI = P*T*R/100)

Week 2:

a. Write a program to swap two variables values with and without using third variable
b. Write a program to find the roots of a quadratic equation.

Week 3:

a. Write a program to find the sum of individual digits of a given positive integer.
b. Write a program, which takes two integer operands and one operator from the user, performs the operation and then prints the result.
(Consider the operators +,-,*, /, % and use Switch Statement)

Week 4:

a. Write a program to find both the largest and smallest number in a list of integers.
b. Write a program to find the sum of integer array elements using pointers

Week 5:

a. Write a program to perform addition of two matrices.
b. Write a program to perform multiplication of two matrices.

Week 6:

a. Write a program to find the length of the string using Pointer.
b. Write a program to count the number of lines, words and characters in a given text.

Week 7:

a. Write a program to find factorial of a given integer using non-recursive function and recursive function.
b. Write program to find GCD of two integers using non-recursive function and recursive function.

Week 8:

a. Write a program using user defined functions to determine whether the given string is palindrome or not.
b. Write a Program to swap the values of two variables using
i) Call by Value ii) Call by Reference

Week 9:

a. Write a program to find the sum of integer array elements using pointers, use dynamic memory allocation to allocate memory.
b. Write a program to perform subtraction of two matrices, Design functions to perform read, display and subtract

Week 10:

a. Write a program to create a structure named book and display the contents of a book.
b. Write a Program to Calculate Total and Percentage marks of a student using structure.

Week 11:

a. Write a program that uses functions to perform the following operations:
i) Reading a complex number ii) Writing a complex number
iii) Addition of two complex numbers iv) Multiplication of two complex numbers
b. Write a program to reverse the first n characters in a file.
(Note: The file name and n are specified on the command line.)

Week 12:

a. Write a program to copy the contents of one file to another.
b. Write a program to merge two files into a third file (i.e., the contents of the first file followed by those of the second are put in the third.

Week 13:

a. Write a program for static implementation of stack
b. Write a program for static implementation of Queue

Week 14:

a. Write a program to perform various operations on singly linked list

Week 15:

a. Write a program for dynamic implementation of stack
b. Write a program for dynamic implementation of Queue

Case Studies

Case 1: Student Record Management System

The main features of this project include basic file handling operations; you will learn how to add, list, modify and delete data to/from file. The source code is relatively short, so thoroughly go through the mini project, and try to analyze how things such as functions, pointers, files, and arrays are implemented.

Currently, listed below are the only features that make up this project, but you can add new features as you like to make this project a better one!

- Add record
- List record
- Modify record
- Delete record

Case 2: Library Management System

This project has 2 modules.

1. Section for a librarian
2. Section for a student

A librarian can add, search, edit and delete books. This section is password protected. That means you need administrative credentials to log in as a librarian.

A student can search for the book and check the status of the book if it is available. Here is list of features that you can add to the project.

1. You can create a structure for a student that uniquely identify each student. When a student borrows a book from the library, you link his ID to Book ID so that librarian can find how a particular book is borrowed.
2. You can create a feature to bulk import the books from CSV file.
3. You can add REGEX to search so that a book can be searched using ID, title, author or any of the field.
4. You can add the student login section.

TEXT BOOKS

1. C Programming and Data Structures, P.Padmanabham, Third Edition, BS Publications
2. Computer programming in C.V.RAjaraman, PHI Publishers.
3. C Programming, E.Balagurusamy, 3rd edition, TMHPublishers.
4. C Programming, M.V.S.S.N Venkateswarlu and E.V.Prasad,S.Chand Publishers
5. Mastering C,K.R.Venugopal and S.R.Prasad, TMH Publishers.

Course Outcomes:

1. Ability to apply solving and logical skills to programming in C language.
2. Able to apply various conditional expressions and looping statements to solve problems associated with conditions.
3. Acquire knowledge about role of Functions involving the idea of modularity.
4. Understand and apply the Concept of Arrays, Strings and Pointers dealing with memory management.
5. Acquire knowledge about basic data structures and their implementation.

INSTRUCTIONS TO STUDENTS

These are the instructions for the students attending the lab:

- Before entering the lab the student should carry the following things (MANDATORY)
    1. Identity card issued by the college.
    2. Class notes
    3. Lab observation book
    4. Lab Manual
    5. Lab Record
- Student must sign in and sign out in the register provided when attending the lab session without fail.
- Come to the laboratory in time. Students, who are late more than 15 min., will not be allowed to attend the lab.
- Students need to maintain 100% attendance in lab if not a strict action will be taken.
- All students must follow a Dress Code while in the laboratory
- Foods, drinks are NOT allowed.
- All bags must be left at the indicated place.
- Refer to the lab staff if you need any help in using the lab.
- Respect the laboratory and its other users.
- Workspace must be kept clean and tidy after experiment is completed.
- Read the Manual carefully before coming to the laboratory and be sure about what you are supposed to do.
- Do the experiments as per the instructions given in the manual.
- Copy all the programs to observation which are taught in class before attending the lab session.
- Students are not permitted to use phones, Flash drives, Internet without permission of lab-in charge.
- Lab records need to be submitted on or before the date of submission.

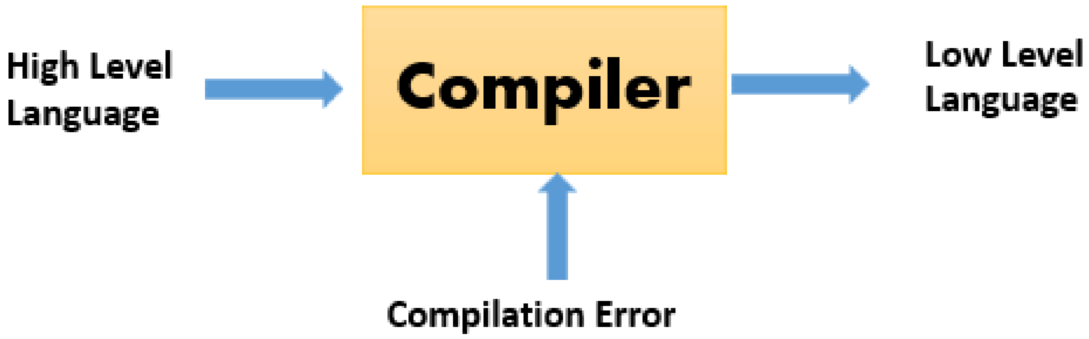
CONTENTS

What are Computer Programming Languages?

Computer programming languages allow us to give instructions to a computer in a language the computer understands. Just as many human-based languages exist, there are an array of computer programming languages that programmers can use to communicate with a computer. The portion of the language that a computer can understand is called a "binary." Translating programming language into binary is known as "compiling." Each language, from C Language to Python, has its own distinct features, though many times there are commonalities between programming languages.

What is a Compiler?

A compiler is a special program that processes statements written in a particular programming language and turns them into machine language or "code" that a computer's processor uses. Typically, a programmer writes language statements in a language such as Pascal or C one line at a time using an editor. The file that is created contains what are called the source statements. The programmer then runs the appropriate language compiler, specifying the name of the file that contains the source statements.



Why use a Compiler?

- Compiler verifies entire program, so there are no syntax or semantic errors
- The executable file is optimized by the compiler, so it is executes faster
- Allows you to create internal structure in memory
- There is no need to execute the program on the same machine it was built
- Translate entire program in other language
- Generate files on disk
- Link the files into an executable format
- Check for syntax errors and data types
- Helps you to enhance your understanding of language semantics
- Helps to handle language performance issues
- Opportunity for a non-trivial programming project
- The techniques used for constructing a compiler can be useful for other purposes as well

C programming language was developed in 1972 by Dennis Ritchie at bell laboratories of AT&T (American Telephone & Telegraph), located in the U.S.A. Dennis Ritchie is known as the founder of the c language. It was developed to overcome the problems of previous languages such as B, BCPL, etc.

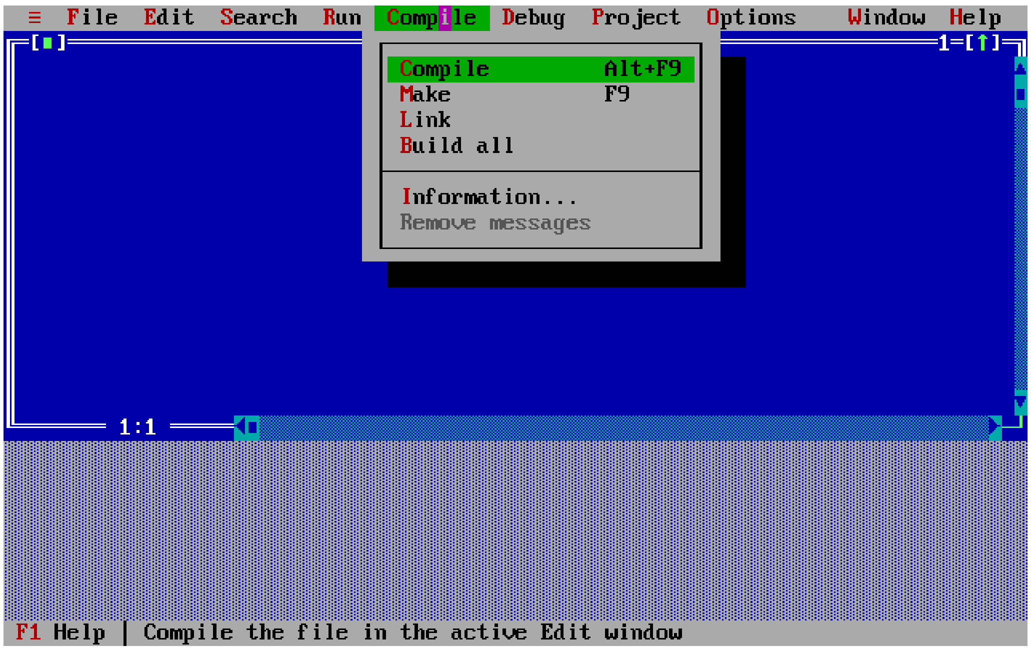
Standardization of C language

Language	Year	Developed By
Algol	1960	International Group
BCPL	1967	Martin Richard
B	1970	Ken Thompson
Traditional C	1972	Dennis Ritchie
K & R C	1978	Kernighan & Dennis Ritchie
ANSI C	1989	ANSI Committee
ANSI/ISO C	1990	ISO Committee
C99	1999	Standardization Committee

Widely used compilers are

1. **Turbo C (16 bit compiler)**
2. **GCC based compilers(32 bit compiler)**

1. Compilation Process in Turbo C In Windows Operating System



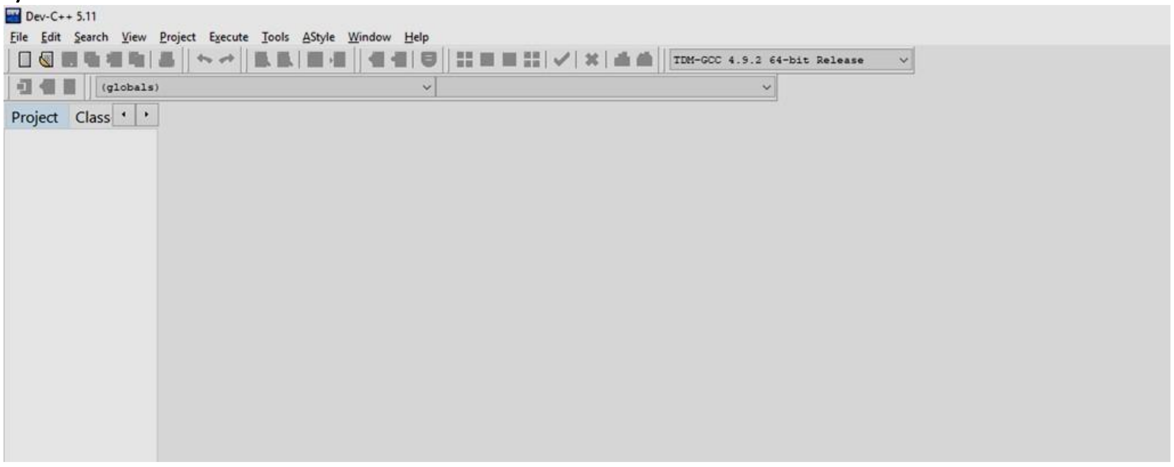
F1 Help | Compile the file in the active Edit window

Open Turbo C Editor.

1. Select **"File"** from Menu bar and select option **"New"**
2. Save C program in filename .C extension.
3. To do compiling – **Select -> Compile** from menu and **click-> compile**.
   If the compilation is success – you will see a **"success"** message. Else you will see the number of errors.
4. To RUN the program – you may **select ->Run** from menu and **click -> Run**
   Now you will see the output screen.

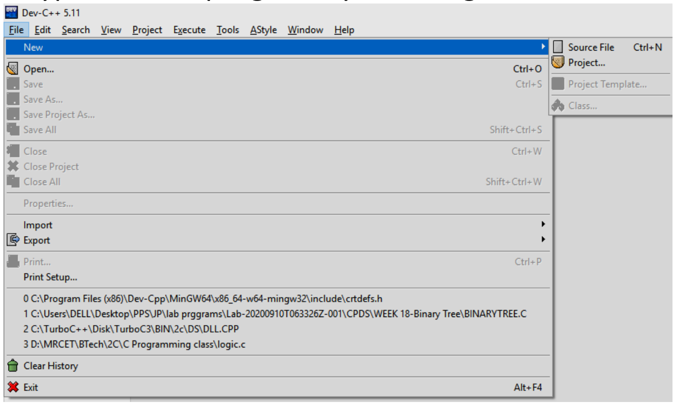
2. Compilation of C program in DEV C++ editor in Windows

Dev-C++ is a free full-featured integrated development environment (IDE) distributed under the GNU General Public License for programming in C and C++ for windows operating system.

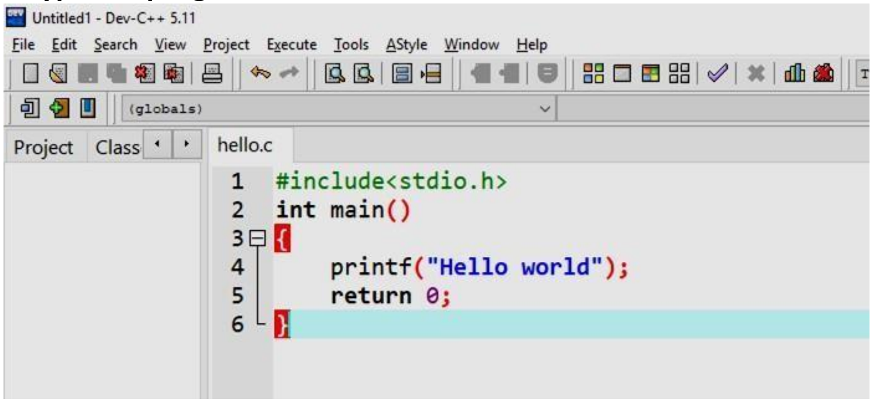

Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Class

Procedure for compilation and execution:

1. Type the new program by selecting new source file from file menu

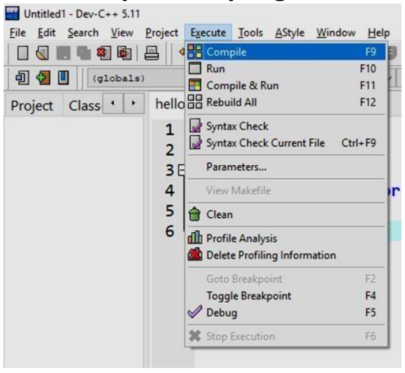

Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
New
Source File Ctrl+N
Project...
Project Template...
Class...
Open... Ctrl+O
Save Ctrl+S
Save As...
Save Project As...
Save All Shift+Ctrl+S
Close Ctrl+W
Close Project
Close All Shift+Ctrl+W
Properties...
Import
Export
Print... Ctrl+P
Print Setup...
0 C:\Program Files (x86)\Dev-Cpp\MinGW64\x86_64-w64-mingw32\include\crtdefs.h
1 C:\Users\DELL\Desktop\PPS\JP\lab prggrams\Lab-20200910T063326Z-001\CPDS\WEEK 18-Binary Tree\BINARYTREE.C
2 C:\TurboC++\Disk\TurboC3\BIN\2c\DS\DLL.CPP
3 D:\MRCET\BTech\2C\C Programming class\logic.c
Clear History
Exit Alt+F4

2. Type the program and save the file with a name

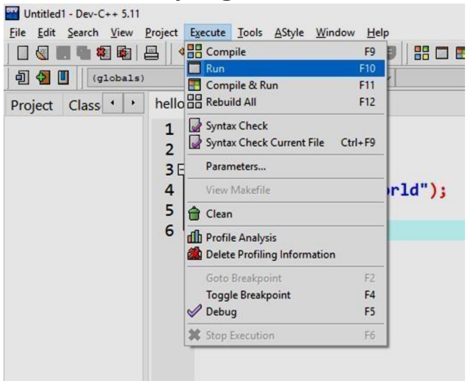

Untitled1 - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Class
hello.c

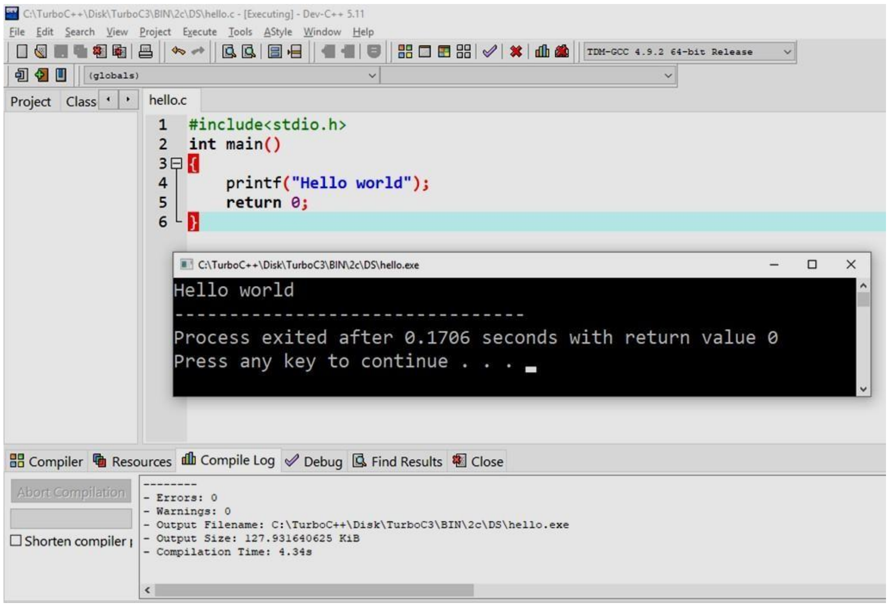
```c
#include<stdio.h>
int main()
{
    printf("Hello world");
    return 0;
}
```

3. To compile the program select compile from execute menu or press F9



4. To run the program select run from execute menu or press F10




C:\TurboC++\Disk\TurboC3\BIN\2c\DS\hello.c - [Executing] - Dev-C++ 5.11
TDM-GCC 4.9.2 64-bit Release
hello.c
C:\TurboC++\Disk\TurboC3\BIN\2c\DS\hello.exe
Compiler
Resources
Compile Log
Debug
Find Results
Close
Abort Compilation
Shorten compiler

```
#include<stdio.h>
int main()
{
    printf("Hello world");
    return 0;
}
```

```
Hello world
--------------------------------
Process exited after 0.1706 seconds with return value 0
Press any key to continue . . .
```

```
--------
- Errors: 0
- Warnings: 0
- Output Filename: C:\TurboC++\Disk\TurboC3\BIN\2c\DS\hello.exe
- Output Size: 127.931640625 KiB
- Compilation Time: 4.34s
```

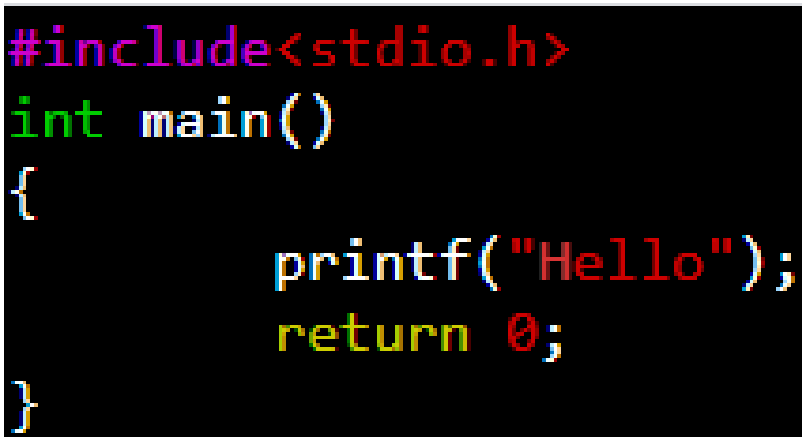
3.Compiling C program using GCC in Linux

Released by the Free Software Foundation. gcc is a Linux-based C compiler usually operated via the command line. It often comes distributed with a Linux installation.
Compilation process of a C program

$gcc filename
The default executable output of gcc is "a.out",
Running the output file

$ ./a.out
It is also possible to specify a name for the executable file at the command line by using the syntax -o outputfile , as shown in the following example : -
gcc filename -o outputfile

Program execution in Linux Environment using vi editor
1.To Type the program use vi editor
[jayapalmedida@webminal.org ~]$vi sample.c
To Type the program press i



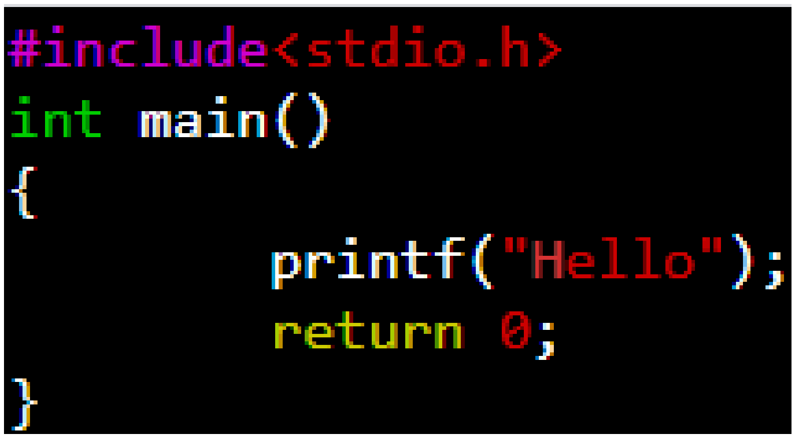
```
#include<stdio.h>
int main()
{
        printf("Hello");
        return 0;
}
```

To save the program

- Press ESC
- press :wq (save and quit)

2. Compilation
[jayapalmedida@webminal.org ~]$gcc sample.c -o output
3. Running the program
[jayapalmedida@webminal.org ~]$./output
Hello

Program execution in Linux Environment using gedit editor
1.To Type the program use gedit editor
[jayapalmedida@webminal.org ~]$gedit sample.c



```
#include<stdio.h>
int main()
{
		printf("Hello");
		return 0;
}
```

Type the program .
Save the program .
Close the editor.

3. Compilation

```
[jayapalmedida@webminal.org ~]$gcc sample.c -o output
```

4. Running the program

```
[jayapalmedida@webminal.org ~]$./output
hello
```

Week 1

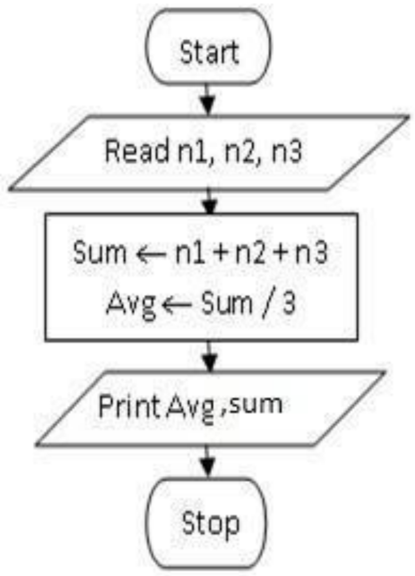
1. a) Write a program to find the sum and average of three numbers.

Aim: Find the sum and average of three numbers

Algorithm:

```
Step 1: Start
Step 2: Read values num1, num2, num3
Step 3: Add num1, num2, num3 and assign the result to sum.
            sum←num1+num2 +num3
            average ← sum/3
Step 4: Display sum and average
Step 5: Stop
```

Flow Chart:



Program:

```
#include<stdio.h>
int  main( )
 {
   int a,b,c;
   int sum,average;
   printf("Enter any three integers: ");
   scanf("%d%d %d",&a,&b,&c);
   sum = a+b+c;
   average=sum/3
   printf("Sum and average of three integers: %d %d",sum,average);
   return 0;
}
```

SAMPLE INPUT:

Enter any three integers: 2 4 5

EXPECTED OUTPUT:

Sum and average of three integers: 11 3

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

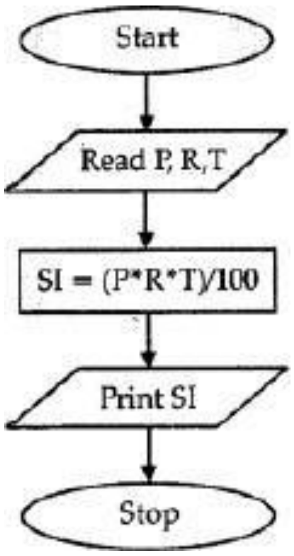
1. b) Write a program to calculate simple interest(SI) for a given principal (P), time (T), and rate of interest (R) (SI = P*T*R/100)

Aim: To find the simple interest

Algorithm:

Step 1: Start.
Step 2 : Read Principal Amount, rate and time.
Step 3 : **Calculate Interest** using formula SI= ((amount*rate*time)/100)
Step 4 : Print **Simple Interest SI**.
Step 5 : Stop

Flow chart



Program:

```
#include<stdio.h>
int main()
{
int p,r,t,si;
        printf("Input principle:");
        scanf("%d",&p);
        printf("Rate of interest:");
        scanf("%d",&r);
        printf("Enter time(in years):");
        scanf("%d",&t);
        si=(p*r*t)/100;
        printf("Simple interest = %d",si);
return 0;
}
```

SAMPLE INPUT:

Input principle: 10000
Rate of interest: 12
Enter time(in years): 2

EXPECTED OUTPUT:

Simple interest = 2400

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise

1.Write a program to find distance when initial velocity, acceleration and time is given.
2.Write a program to find compound interest.
3. Write a program to find amount of memory required by different types of variables.
4. Write a program to evaluate algebraic expression (ax+b)/(ax-b).
5.Write a program to multiply number by 2 using shift operator.

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 2

2 a)Write a program to swap two variables values with and without using third variable

AIM: To swap two variable values using a third variable

DESCRIPTION:

Swap the values of the variable using temporary variable t

t = a
a = b
b = t

ALGORITHM:

- **using a third variable**

```
Step 1 : Start
Start 2 : READ num1, num2
Start 3 : temp = num1
Start 4 : num1 = num2
Start 5 : num2 = temp
Start 6 : PRINT num1, num2
Start 7 : Stop
```

- **without using a third variable**

```
Step 1 : Start
Start 2 : READ num1, num2
Start 3 : num1 = num1 + num2
Start 4 : num2 = num1 - num2
Start 5 : num1 = num1 - num2
Start 6 : PRINT num1, num2
Start 7 : Stop
```

FLOWCHART:

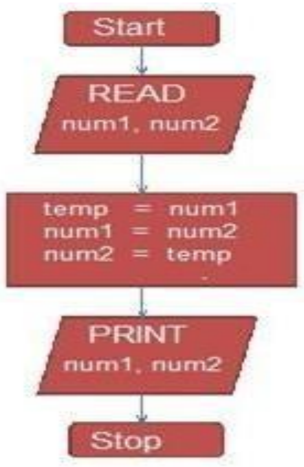


fig a : using a third variable

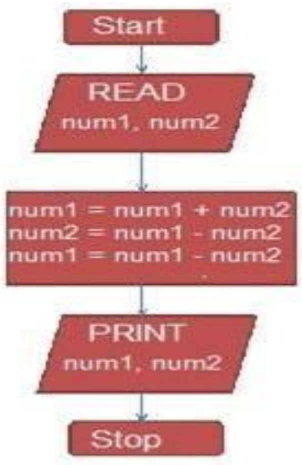


fig b : without using a third variable

PROGRAM:

using a third variable

```
#include<stdio.h>
 int  main()
{
      int x, y, t;
                  printf("Enter two integers: ");
                  scanf("%d%d", &x, &y);
                  printf("Before Swapping\nFirst integer = %d\nSecond integer = %d\n", x, y);
                  t = x;
                 x = y;
                 y = t;
                 printf("After Swapping\nFirst integer = %d\nSecond integer = %d\n", x, y);
     return 0;
}
```

SAMPLE INPUT:

```
Enter two integers: 10  20
```

EXPECTED OUTPUT:

```
Before Swapping
First integer = 10
Second integer = 20
After  Swapping
First integer = 20
Second integer = 10
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

PROGRAM: Without using a third variable

```
#include <stdio.h>
 int main()
{
  int a, b;
            printf("Enter two integers: ");
             scanf("%d%d", &a, &b);
             printf("Before Swapping\nFirst integer = %d\nSecond integer = %d\n", a, b);
             a = a + b;
             b = a - b;
             a = a - b;
           printf("After Swapping\nFirst integer = %d\nSecond integer = %d\n", a, b);
  return 0;
}
```

SAMPLE INPUT:

```
Enter two integers: 23 45
```

EXPECTED OUTPUT:

```
Before Swapping
First integer = 23
Second integer = 45
After  Swapping
First integer = 45
Second integer = 23
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

2 b) Write a program to find the roots of a quadratic equation.

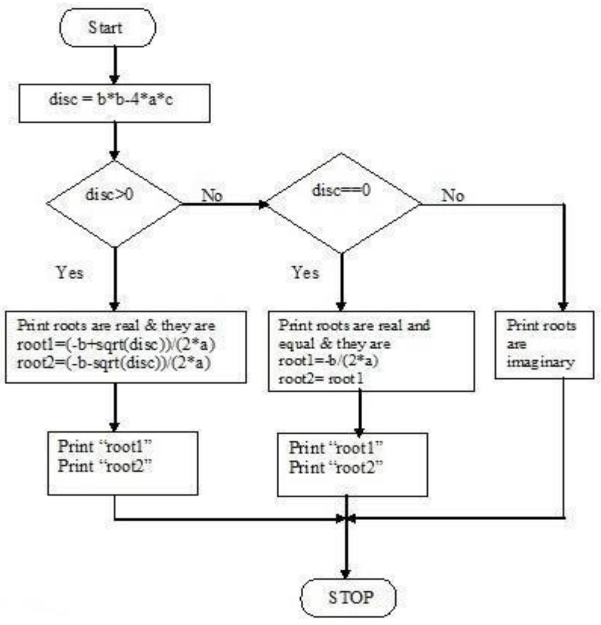
AIM: To find the roots of a quadratic equation.

Description: roots of quadratic equation are $\frac{-b\pm\sqrt{b^2-4ac}}{2a}$

ALGORITHM:

```
Step 1: Start
Step 2: Read a,b,c
Step 3: calculate disc = b*b-4*a*c
Step 4: if(disc>0)
          Begin
Step 5:   root1=(-b+sqrt(disc))/(2*a)
Step 6: root2=(-b-sqrt(disc))/(2*a)
Step 7: Print “Root1” , “Root2”
         End
Step 8: else if(disc=0)
         Begin
Step 9: root1=-b/(2*a)
Step 10: root2=root1;
Step 11: Print “Root1” , “Root2”
         End
Step 12: else
Step 13: Print Roots are imaginary
Step 14: Stop
```

Flow Chart


Start
disc = b*b-4*a*c
disc>0
No
Yes
disc==0
No
Yes
Print roots are real & they are
root1=(-b+sqrt(disc))/(2*a)
root2=(-b-sqrt(disc))/(2*a)
Print roots are real and equal & they are
root1=-b/(2*a)
root2= root1
Print roots are imaginary
Print “root1”
Print “root2”
Print “root1”
Print “root2”
STOP

PROGRAM:

```
#include<stdio.h>
#include<math.h>
int main()
{
int a,b,c;
float disc, root1, root2;
float img,real;
        printf("ENTER VALUES FOR a,b,c:\n");
        scanf("%d%d%d",&a,&b,&c);
        disc=(float)b*b-4*a*c;
        if(disc>0)
        {       printf("THE ROOTS ARE REAL & UNEQUAL:\n");
                root1=(-b+sqrt(disc))/(2*a);
                root2=(-b-sqrt(disc))/(2*a);
                printf("Root1=%f\n",root1);
                printf("Root2=%f\n",root2);
        }
        else if(disc==0)
        {               printf("THE ROOTS ARE REAL AND EQUAL:\n");
                        root1=-b/(2*a);
                        root2=root1;
                        printf("Root1=%f\n",root1);
                        printf("Root2=%f\n",root2);
        }
        else
        {
          printf("THE ROOTS ARE IMAGINARY:\n");
        }
return 0;
}
```

SAMPLE INPUT:

```
        ENTER VALUES FOR a, b, c
        1       4       4
```

EXPECTED OUTPUT:

```
        THE ROOTS ARE EQUAL AND THEY ARE..     Root1=-2    Root2=-2
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise:

1) Write a program to swap two variables values without using third variable using XOR(^) operator.
2) Write a program to check whether the entered year is leap year or not (a year is leap if it is divisible by 4 and divisible by 100 or 400)
3) Write a program to Check whether given number is even or odd through command line
4) Write a program to find largest of three numbers
5) Write a program to determine whether an input number is even or odd

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 3

3 a)Write a C program to find the sum of individual digits of a given positive integer.

AIM: To find the sum of individual digits of positive integer.

Description:

Summation of digits of a number

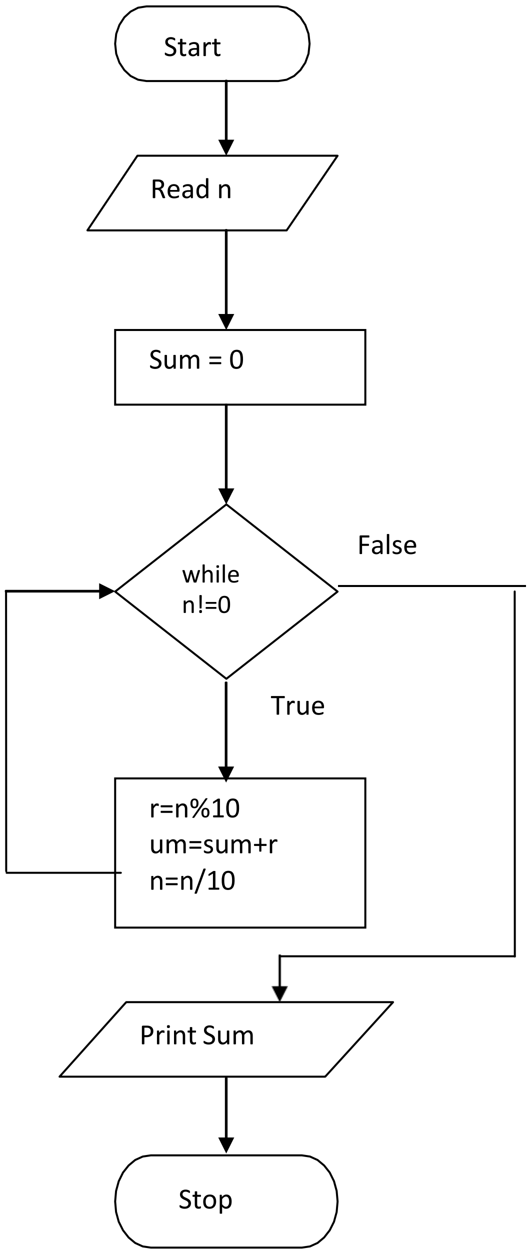
Ex: 1234

Summation =1+2+3+4=10

ALGORITHM:

```
Step 1: Start
Step 2: Read n
Step 3: Initialize sum ← 0
Step 4: while(n!=0)
        Begin
Step 5: r←n%10
Step 6: sum← sum + r
Step 7: n←n/10
        End
Step 8: Print “sum”
Step 9: Stop
```

FLOWCHART:



PROGRAM:

```
#include<stdio.h>
int  main()
{
int n,r,sum=0;
        printf("ENTER A POSITIVE INTEGER \n");
        scanf("%d",&n);
        while(n!=0)
        {
                r=n%10;
                sum=sum+r;
                n=n/10;
        }
        printf("THE SUMOF INDIVIDUAL DIGITS OF A POSITIVE INTEGER IS..%d", sum);
return 0;
}
```

SAMPLE INPUT:

```
        ENTER A POSITIVE INTEGER
        5 3 2 1
```

EXPECTED OUTPUT:

```
        THE SUM OF INDIVIDUAL DIGITS OF A POSITIVE INTEGER IS..11
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

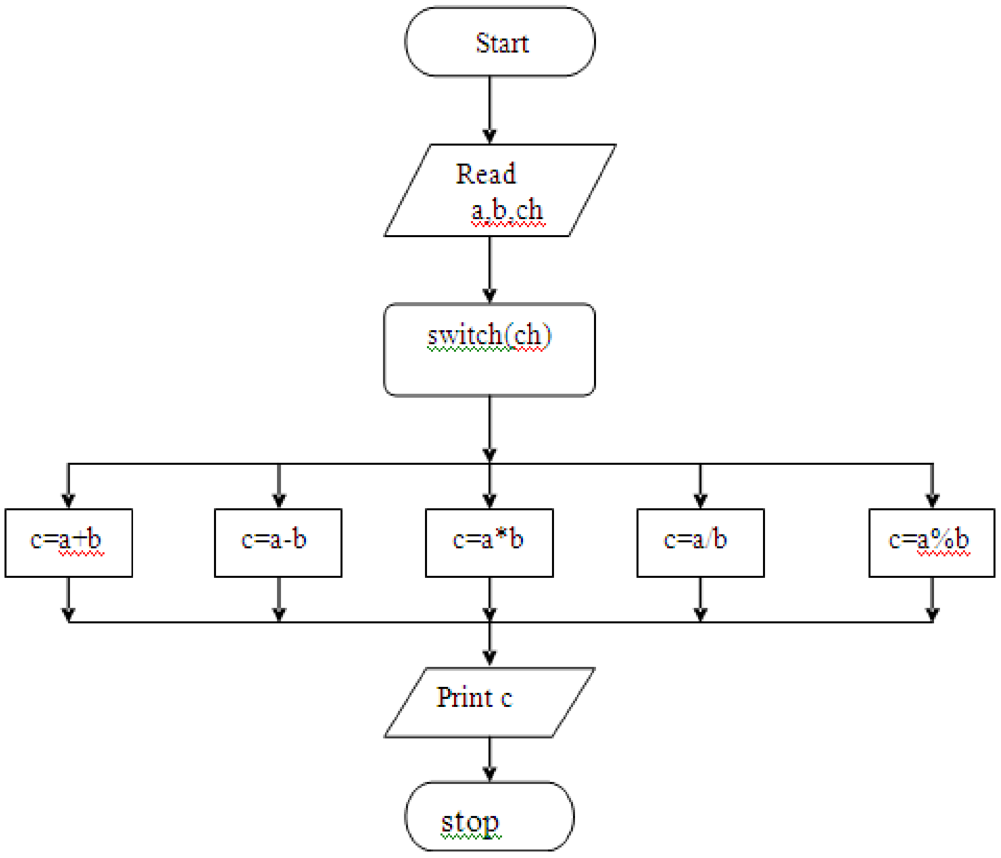
3 b) Write a C program, which takes two integer operands and one operator from the user, performs the operation and then prints the result. (Consider the operators +,-,*, /, % and use Switch Statement)

AIM: To perform arithmetic operations using switch statement.

Algorithm:

```
Step 1: Read a,b
Step 2: Print “Menu Options”
Step 3: do
Begin Step 4: Read ch
Step 5: switch(ch)
Begin Step 6:
case ‘+’: Begin
          Calculate c = a+b
           Print “c”
          break;
          End
case ‘-‘: Begin
           Calculate c = a-b
           Print “c”
           break;
           End
case ‘*’:Begin
           Calculate c = a*b
           Print “c”
           break;
           End
case ‘/’:Begin
           Calculate c = a/b
           Print “c”
           break;
           End
case ‘%’:Begin
           Calculate c = a%b
           Print “c”
           break;
           End
default:
           Print “Invalid choice”
           End
```

Flowchart



Program:

```
#include<stdio.h>
int  main()
{
int a,b,c;
char ch,t;
 printf("ENTER TWO VALUES FOR a & b\n");
            scanf("%d %d",&a,&b);
            scanf(“%c”,&t);                // to skip the newline character
             printf("MENU OPTIONS \n");
             printf("************\n");
             printf("Addition\n");
             printf("Subtraction\n");
             printf("Multiplication\n");
             printf("Division\n");
             printf("Modulus\n");
             printf("\n");
             printf("ENTER Operator : \n");
```

```
        scanf("%c",&ch);
                switch(ch)
                {
                        case '+':c=a+b;
                                printf("The addition of %d and %d is..%d\n",a,b,c); break;
                        case '-':c=a-b;
                                printf("The subtraction of %d and %d is..%d\n",a,b,c); break;
                        case '*':c=a*b;
                                printf("The multiplication of %d and %d is..%d\n",a,b,c); break;
                        case '/':c=a/b;
                                printf("The division of %d and %d is..%d\n",a,b,c); break;
                        case '%':c=a%b;
                                printf("The modulus of %d and %d is..%d\n",a,b,c); break;
                        default:printf("INVALID CHOICE\n"); }
                }
return 0;
}
```

SAMPLE INPUT:

```
        ENTER TWO VALUES FOR a & b: 20     16
```

EXPECTED OUTPUT:

```
        MENU OPTIONS
        *************
        Addition
        Subtraction
        Multiplication
        Division
        Modulus

        ENTER Operator :  +
The addition of 20 and 16 is..36
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise:

1) Write a program to generate the first n terms of the Fibonacci using all loops.
2) Write a program to check whether given number is Armstrong number or not.
3) Write a program to find HCF and LCM using recursion.
4) Write a program to generate prime numbers from 1 to n.
5) Write a program to Check for a vowel using switch statement.

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 4

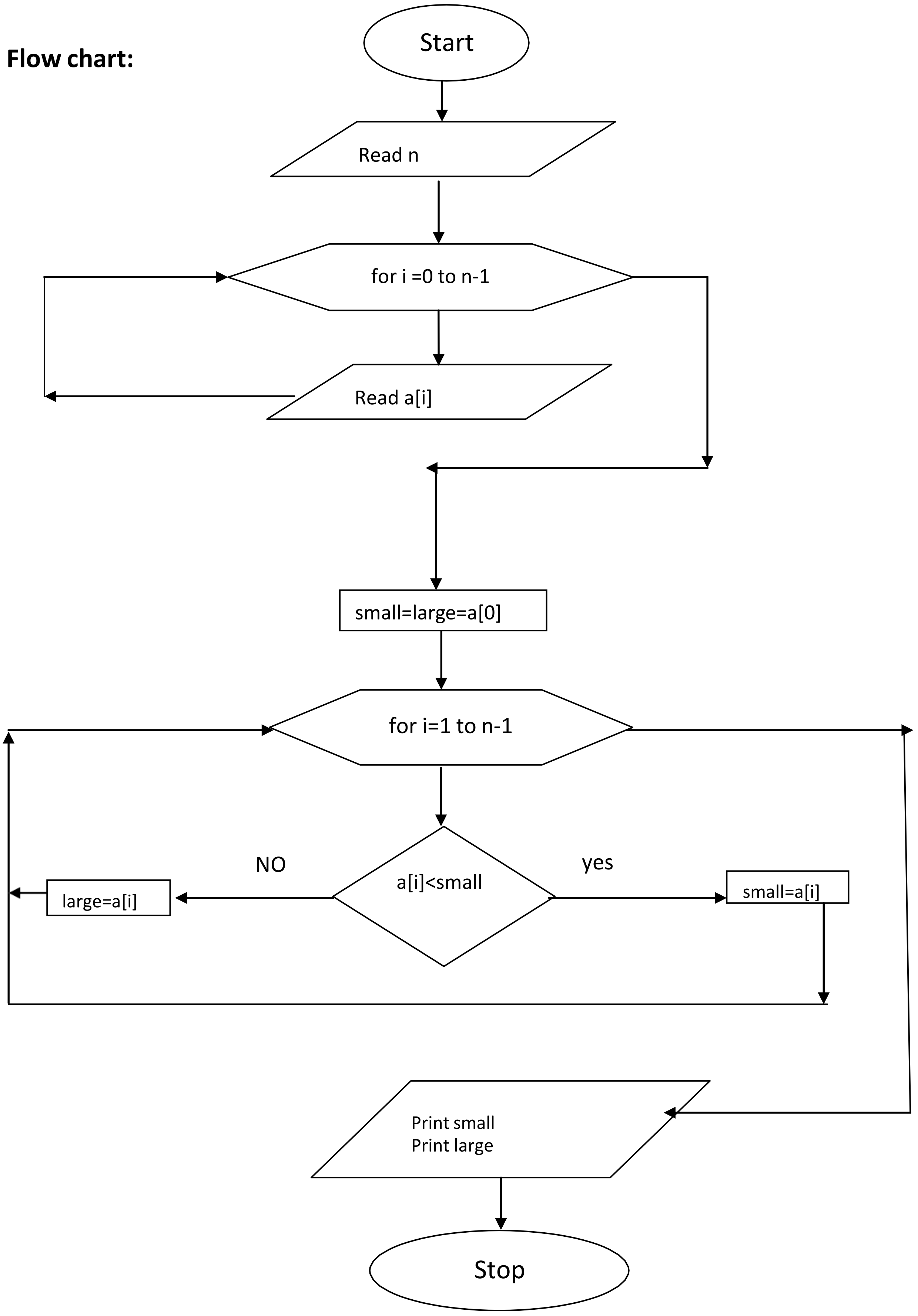
4 a) Write a program to find both the largest and smallest number in a list of integers

AIM: To find the largest and smallest number in a list of integers.

ALGORITHM:

```
Step 1: start
Step 2: read n
Step 3: initialize i=0
Step 4: if i<n do as follows. If not goto step 5
            Read a[i]
            Increment i
           goto step 4
Step 5: small=a[0], large=a[0]
Step 6: initialize i=0
Step 7: if i<n do as follows. If not goto step 8
            If a[i]<small
                Assign small=a[i]
            If a[i]>large
                Assign large=a[i]
        Increment i goto Step 7
Step 8: print small, large
Step 9: stop
```

Flow chart:


Start
Read n
for i =0 to n-1
Read a[i]
small=large=a[0]
for i=1 to n-1
a[i]<small
NO
yes
large=a[i]
small=a[i]
Print small
Print large
Stop

Program:

```
#include<stdio.h>
int main()
{
 int a[10],i,n,small,large;
            printf("Enter The Array Size:");
            scanf("%d",&n);
            printf("Enter The Array elements:");
            for(i=0;i<n;i++)// read the elements of an array
            scanf("%d",&a[i]);
            small=a[0];
            large=a[0];
            for(i=1;i<n;i++)// read the elements of an array
            {
                    if(a[i]<small)// check the condition for minimum value
                        small=a[i];
                    if(a[i]>large)//check the condition for maximum value
                                large=a[i];
            }
          printf("largest value is:%d\n",large);
          printf("smallest value is:%d\n",small);
          return 0;
}
```

SAMPLE INPUT:

```
            Enter The Array Size: 10
            ENTER THE ELEMENTS OF ARRAY
            7      10      9      8      6      5      2      3      4      1
```

EXPECTED OUTPUT:

```
                  largest value is : 10
                 smallest value is : 1
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

4 b) Write a C program to find the sum of integer array elements using pointers

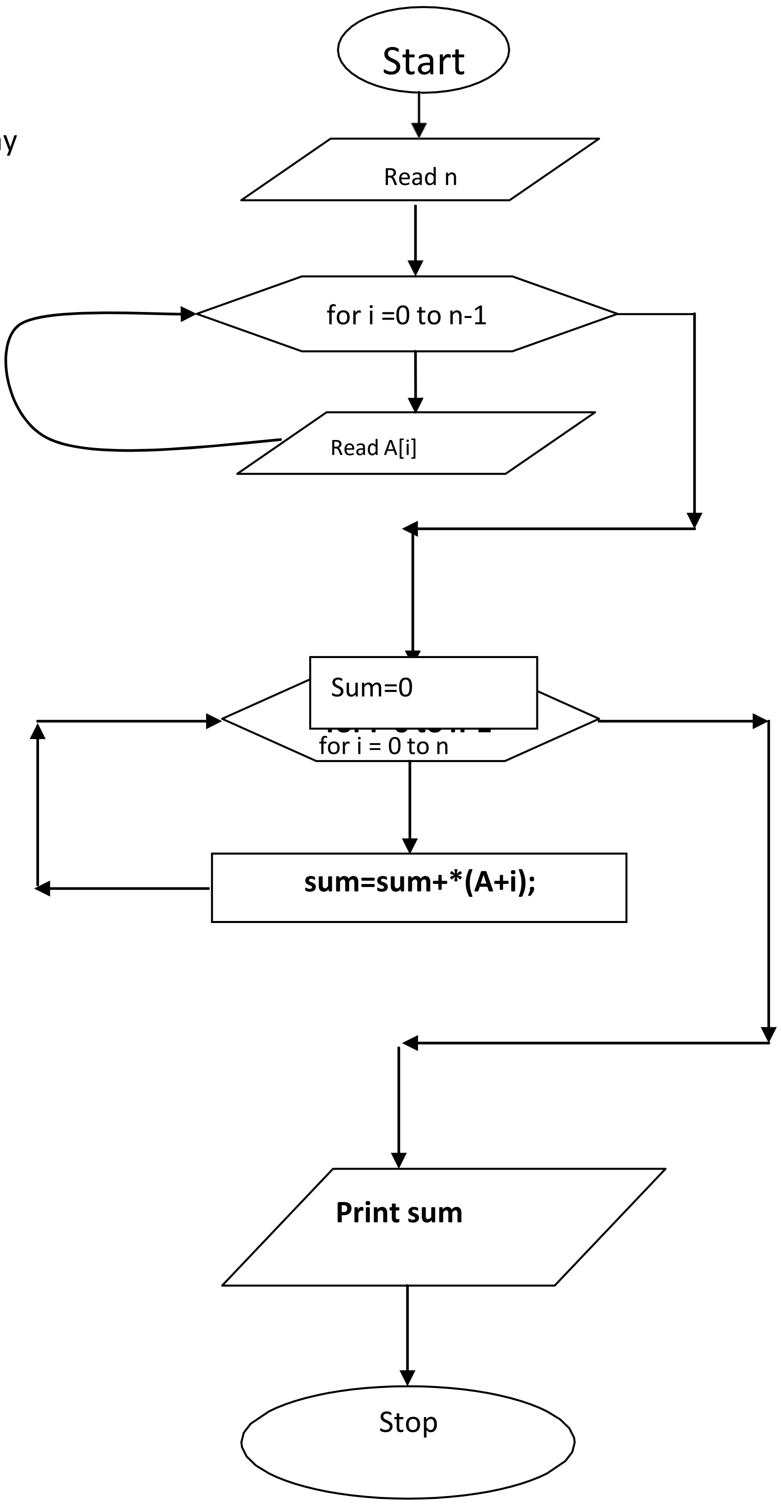
AIM: To find the sum of integer array elements using pointers.

Description: Consider an integer array. By considering the name of the array as initial address of the first element, addition will be performed.

ALGORITHM:

```
Step 1:start
Step 2: Read n elements into array
Step 3: initialize sum=0
Step 4: for(i=0;i<n;i++)
Step 5: sum=sum+*(A+i)
step 6:print sum
step 7: stop
```

Flow chart:



PROGRAM:

```
 #include<stdio.h>
int  main( )
{
int A[50],sum=0,i,n;
          printf("Enter how many values to read");
          scanf("%d",&n);
          printf("enter elements into array");
          for(i=0;i<n;i++)
          scanf("%d",&A[i]);
          for(i=0;i<n;i++)
          sum=sum+*(A+i);
          printf("the addition of array elements is %d",sum);
          return 0;
}
```

SAMPLE INPUT:

```
Enter the array elements : 1 2 3 4 5 6 7 8 9 1
```

EXPECTED OUTPUT:

```
the addition of array elements is  : 46
```

Record at least 2 results

Assessment	
Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise

1) Write a C program to generate the first n terms of the Fibonacci, use one dimensional array to store the series.
2) Write a program to search for a given element in an array using linear search.
3) Write a program to find Fibonacci prime numbers.
4) Write a C Program to Sort the Array in an Ascending Order.
5) Write a program to count a total number of duplicate elements in an array.

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 5

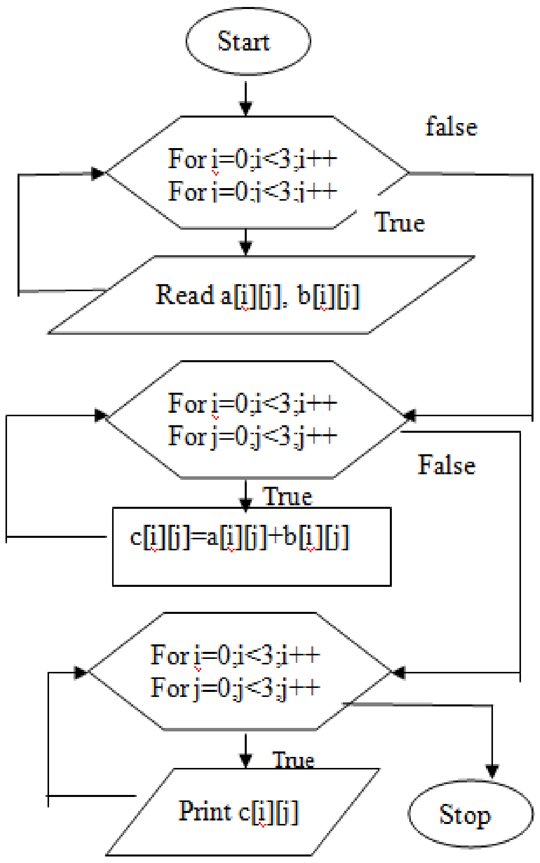
5 a)Write a program to perform addition of two matrices.

AIM: To perform addition of two matrices.
Description: Consider two matrices and their order is R1xC1 and R2XC2. The condition is R1==R2 and C1==C2, then only the addition is possible.

ALGORITHM:

```
Step 1: Start
Step 2: Read the order of the two matrices R1, C1 and R2, C2
Step 3: if R1 != R2 or C1 != C2
            Print "Addition not possible"
            goto step 8
Step 4: for i is 0 to R1 by step 1
            for j is 0 to C1 by step 1
                read a[i][j]
Step 5: for i is 0 to R2 by step 1
            for j is 0 to C2 by step 1
                read b[i][j]
Step 6: for i is 0 to R1 by step 1
            for j is 0 to C1 by step 1
                calculate
c[i][j]=a[i][j]+b[i][j]
Step 7: for i is 0 to R1 by step 1
          for j is 0 to C1 by step 1
                print c[i][j]
Step 8: Stop
```

Flowchart:



PROGRAM:

```
#include<stdio.h>
int main()
{
int a[5][5],b[5][5],c[5][5];
int i,j,p,q,r,s;
        printf("ENTER ORDER OF A MATRIX\n");
        scanf("%d%d",&p,&q);
        printf("ENTER ORDER OF B MATRIX\n");
        scanf("%d%d",&r,&s);
        if(p==r&&q==s)
        {
                printf("ENTER A MATRIX\n");
                for(i=0;i<p;i++)
                        for(j=0;j<q;j++)
                                scanf("%d",&a[i][j]);
                printf("ENTER B MATRIX\n");
                for(i=0;i<p;i++)
                        for(j=0;j<q;j++)
                                scanf("%d",&b[i][j]);
                        for(i=0;i<p;i++)
                                for(j=0;j<q;j++)
                                        c[i][j]=a[i][j]+b[i][j];
                printf(" After Addition of two matrices :\n");
                for(i=0;i<p;i++)
                {
                        for(j=0;j<q;j++)
                        {
                                printf("%d\t",c[i][j]);
                        }
                        printf("\n");
                }
        }
        else
        {
                printf("Addition not possible");
        }
return 0;
}
```

SAMPLE INPUT:

```
ENTER ORDER OF A MATRIX 2 2
ENTER ORDER OF B MATRIX 2 2
ENTER  A MATRIX
1 2
3 4
ENTER B MATRIX
1 2
3 4
```

EXPECTED OUTPUT:

```
 After Addition of two matrices :
2      4
6      8
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

5 b)Write a C program to perform multiplication of two matrices.

AIM: To perform multiplication of two matrices.

Description: Consider two matrices and their order is R1xC1 and R2XC2. The condition is C1==R2 ,then only the multiplication is possible.

ALGORITHM:

```
Step 1: Start
Step 2: Read order of two matrices R1, C1 and R2, C2
Step 3: if C1!=R2
            print "Multiplication not possible"
            goto Step 8
Step 4: for i is 0 to R1 by step 1
            for j is 0 to C1 by step 1
                read a[i][j]
Step 5:  for i is 0 to 21 by step 1
            for j is 0 to C2 by step 1
                read b[i][j]
Step 6:  for i is 0 to R1 by step 1
            for j is 0 to C2 by step 1
                c[i][j]=0;
                for k is 0 to C1 by step 1
                    calculate c[i][j]=c[i][j]+a[i][k]*b[k][j]
Step 7: for i is 0 to R1 by step 1
            for j is 0 to C2 by step 1
                print c[i][j]
Step 8: Stop
```

Program:

```
#include<stdio.h>
int  main()
{ int a[5][5],b[5][5],c[5][5],m,n,p,q;
int i,j,k;
        printf("Enter the size of A Mtrix (Row and Col): \n");
        scanf("%d%d",&m,&n);
        printf("Enter the size of B Mtrix (Row and Col): \n");
        scanf("%d%d",&p,&q);
        if(n!=p)
        {       printf("Multiplication Not Possible\n Please re-enter\n");
                printf("correct size and try again ..... \n");
        }
        else
        {       printf("Enter Matrix A Values Row by Row\n");
                for (i=0;i<m;i++)
                        for(j=0;j<n;j++)
```

```
                                scanf("%d",&a[i][j]);
                    printf("Enter Matrix B Values Row by Row\n");
                    for (i=0;i<p;i++)
                           for(j=0;j<q;j++)
                                scanf("%d",&b[i][j]);
                    //logic for multiplication
                    for (i=0;i<m;i++)
                      {
                           for(j=0;j<q;j++)
                          {
                                c[i][j]=0;
                                for(k=0;k<n;k++)
                                      c[i][j]+= a[i][k]*b[k][j];
                          }
                    }
                    printf("A Matrix is :\n");
                    for (i=0;i<m;i++)
                    {
                          for(j=0;j<n;j++)
                          {
                                printf("%5d",a[i][j]);
                          }
                           printf("\n");
                     }
                 printf("B Matrix is :\n");
                 for (i=0;i<p;i++)
                 {
                    for(j=0;j<q;j++)
                    {
                     printf("%5d",b[i][j]);
                    }
                     printf("\n");
                 }
                 printf("C Matrix is :\n");
                     for (i=0;i<m;i++)
                    {
                          for(j=0;j<q;j++)
                          {
                                 printf("%5d",c[i][j]);
                          }
                           printf("\n");
                     }
               }
         return 0;
```

```
}
```

SAMPLE INPUT:

```
Enter the size of A Mtrix (Row and Col):   2   2
Enter the size of B Mtrix (Row and Col):   2    2
Enter Matrix Value Row by Row
     1   0
     2   6
Enter Matrix Value Row by Row
     3   4
     4   2
```

EXPECTED OUTPUT:

```
A matrix is:
     1   0
     2   6
B Matrix is:
     3   4
     4   2
C matrix is:
     2   4
     24  20
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise

1. Write a program to find whether given matrix is symmetric or not.
2. Write a Program that uses functions to perform transpose of a given Matrices.
3. Write a program to find sum of rows and columns of a Matrix.

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 6

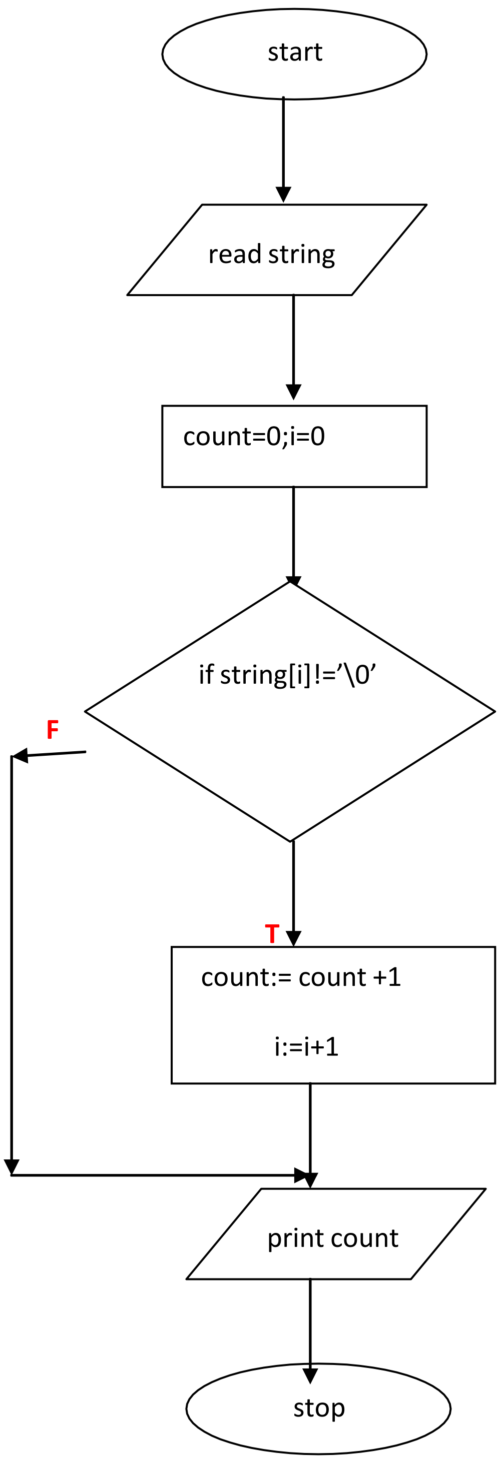
6 a)Write a program to find the length of the string using Pointer.

AIM: To find the length of the string using Pointer.

Algorithm:

```
Step 1:start
Step 2:read  string
Step 3: count=0;i=0
Step 4: if string[i]!='\0'
          count:= count +1
           i:=i+1
           goto step 4
step 5:print count
step 6 stop
```

Flowchart:



Program:

```
#include<stdio.h>
 int string_Len(char*);
int main()
{
   char str[20];
   int length;
              printf("\nEnter any string : ");
                gets(str);
                length = string_Len (str);
                printf("The length of the given string %s is : %d", str, length);
               return 0;
}
 int string_Len (char *p) /* p=&str[0] */
{
   int count = 0;
               while (*p != '\0')
             {    count++;
                   p++;
             }
   return count;
}
```

SAMPLE INPUT:

```
Enter the String : pritesh
```

EXPECTED OUTPUT:

```
Length of the given string pritesh is : 7
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

6b)Write a C program to count the number of lines, words &characters in a given text.

AIM:

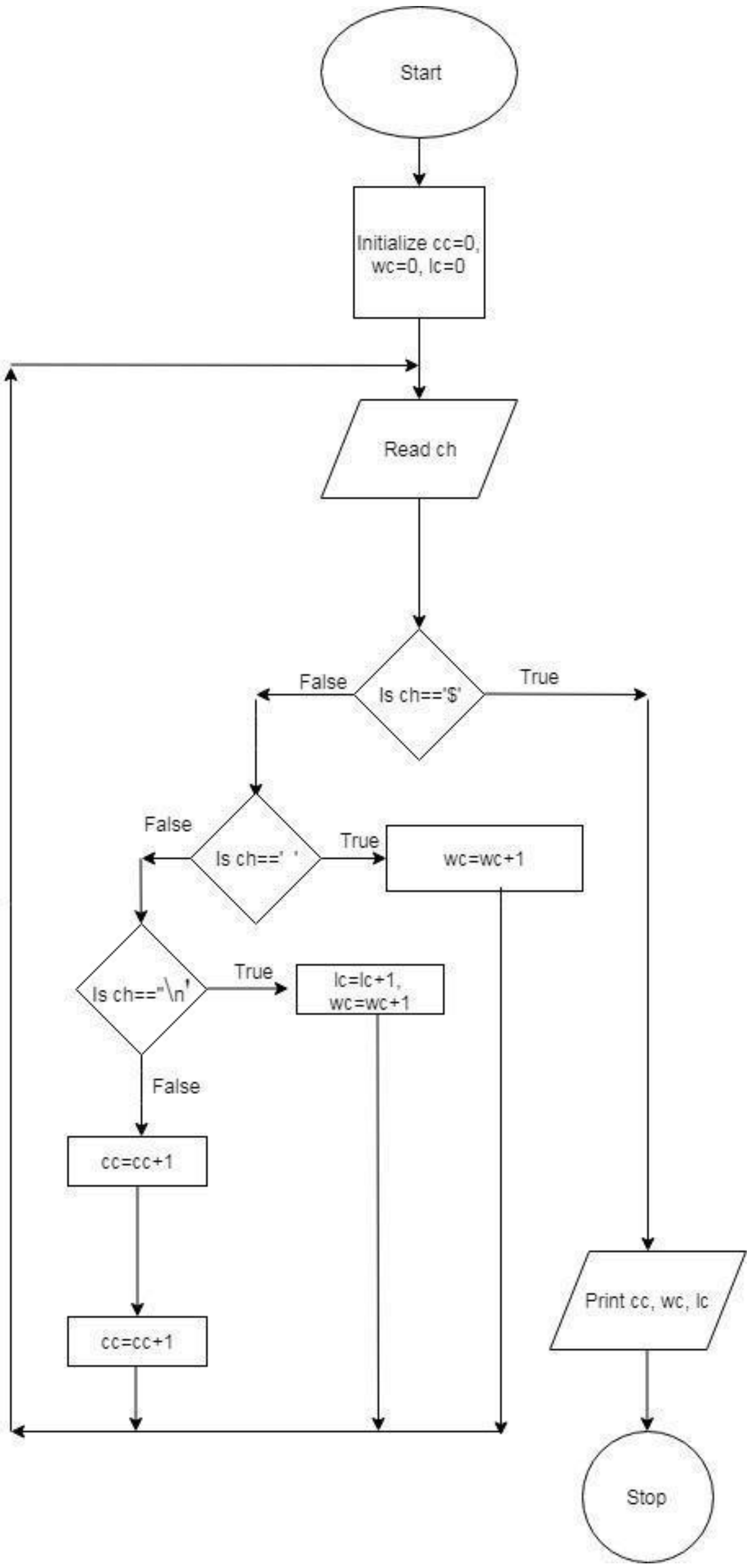
To count the number of lines, words and characters in a given list.

ALGORITHM:

Algorithm

```
Step 1: Initialize charactercount, wordcount and linecount to 0.
Step 2: Read a character ch
Step 3: check if given character is $, if so goto step 7
Step 4: otherwise, if ch = ' ' , increment wordcount by 1, goto step 2
Step 5: else if ch = '\n' (newline character) increment linecount and
                wordcount by 1, goto step 2
Step 6: else increment charactercount by 1, goto step 2
Step 7: Print charactercount, wordcount and linecount
Step 8: Stop
```

Flow Chart:


Start
Initialize cc=0, wc=0, lc=0
Read ch
Is ch=='$'
False
True
Is ch==' '
False
True
wc=wc+1
Is ch=="\n'
True
lc=lc+1, wc=wc+1
False
cc=cc+1
cc=cc+1
Print cc, wc, lc
Stop

PROGRAM:

```
 #include <stdio.h>
int main()
{
   char ch;
   int i=0,wc=0,lc=0,cc=0;
   printf("Enter text at end $");
   while((ch=getchar())!='$')
   {
      if(ch==' ')
         wc++;
      else if(ch=='\n')
      {
         lc++;
         wc++;
      }
      else
         cc++;

   }
   printf("No. of Characters : %d\n",cc);
   printf("No. of Words : %d\n",wc);
   printf("No. of Lines : %d\n",lc);

   return 0;
}
```

SAMPLE INPUT:

```
Enter text at end $abc def
ghi jkl
mno pqr
$
```

EXPECTED OUTPUT:

```
No. of Characters : 18
No. of Words : 6
No. of Lines : 3
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise:

1. Write a program to find length of a string without using library functions

2. Write a program to use function to insert a sub-string in to given main string from a given position.

3. Write a program to compare two strings without using library functions.

4. Write a program to concatenate two strings without using library functions.

5. Write a program to convert lowercase string into upper case without using library functions.

6. Write a program to convert upper case string into lower case without using library functions.

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 7

7 a) Write a program to find factorial of a given integer using non-recursive function and recursive function.

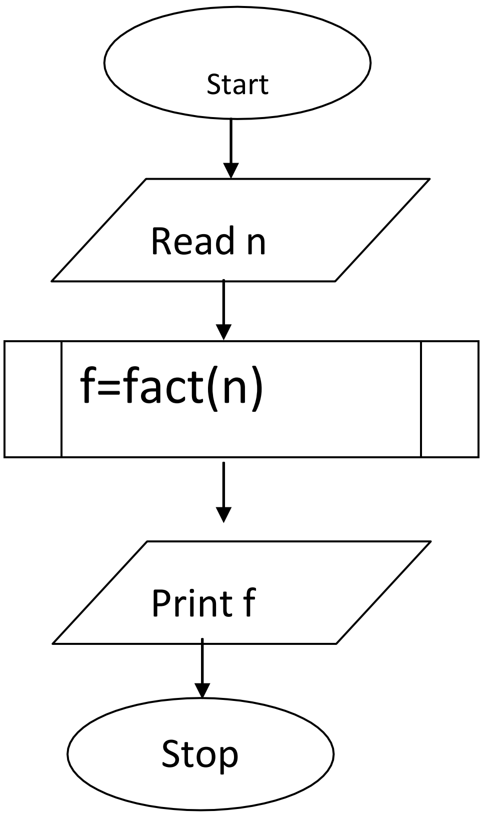
AIM: To find the factorial of a given number using non-recursive function

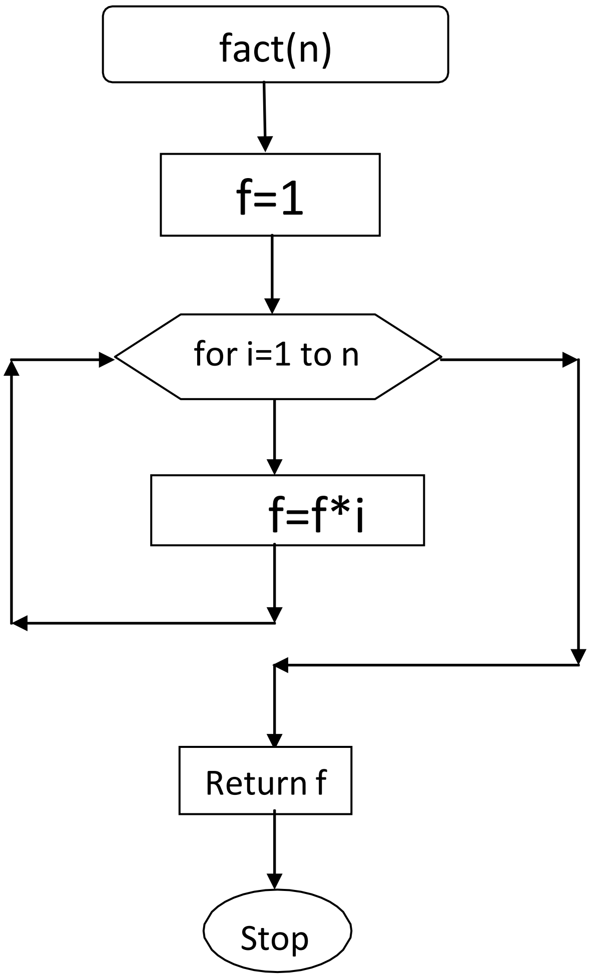
Description: n!=n*(n-1)*(n-2).....*1

ALGORITHM:

```
Step 1: Start
Step 2: Read n
Step 3: Call fact(n) goto step 6
Step 4: Store result in “f”
Step 5: Print “f” goto step 10
Step 6:  Begin                //sub program
         Initialize f ← 1
Step 7: for i is 1 to n by step 2
Step 8: Calculate f = f*i
Step 9: return “f”
         End
Step 10: Stop
```

FLOWCHART:





PROGRAM: write a program for factorial of a given integer using non-recursive function.

```
#include<stdio.h>
int fact(int);
int  main()
{
int n,i,f;
        printf("ENTER A VALUE FOR n:");
        scanf("%d",&n);
        f=fact(n);
        printf("THE FACTORIAL OF A GIVEN NO IS..%d",f);
return 0;
}
int fact(int n)
{
int i,f=1;
        for(i=1;i<=n;i++)
        f=f*i;
return(f);
}
```

SAMPLE INPUT:

```
ENTER A VALUE FOR n: 5
```

EXPECTED OUTPUT:

```
THE FACTORIAL OF A GIVEN NUMBER IS..120
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

AIM: To find the factorial of a given integer using recursive function.

Description: n!=n*(n-1)*(n-2)…..*1

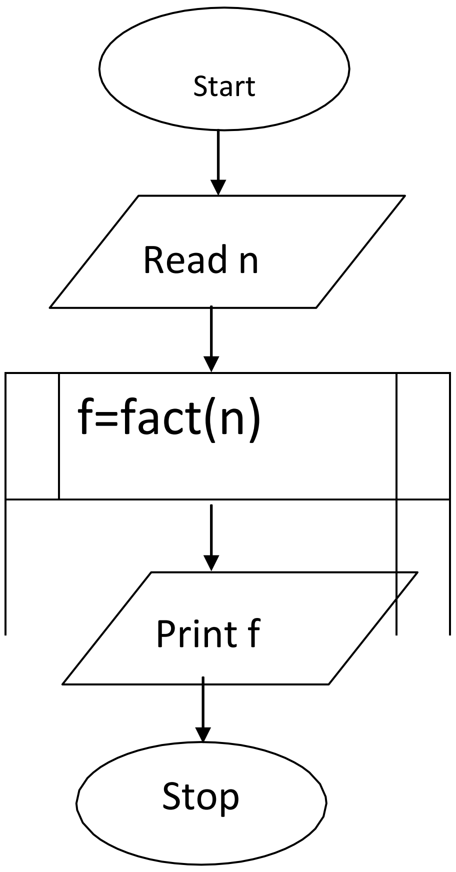
ALGORITHM:

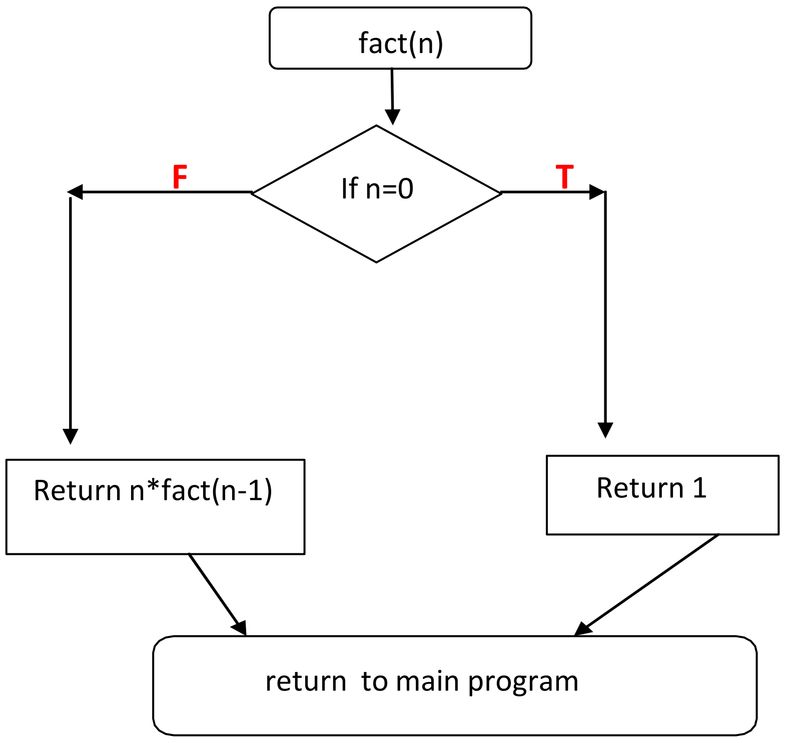
```
Step 1: start
Step 2: read n
Step 3: call sub program f=fact(n)
Step 4: print the f value
Step 5: stop
```

Sub program fact(n):

```
Step 1: if n=0 return 1 to main program
Step 2: return n*fact(n-1) to main program
```

FLOWCHART:





PROGRAM:

```
#include<stdio.h>
int fact(int);
int main()
{
int n,res;
        printf("ENETR A NUMBER:");
        scanf("%d",&n);
        res=fact(n);
        printf("THE FACTORIAL OF A GIVEN NUMBER IS..%d",res);
return 0;
}
int fact(int n)
{
if(n==0)
        return(1);
else
        return(n*fact(n-1));
}
```

SAMPLE INPUT:

```
ENTER A VALUE FOR n 5
```

EXPECTED OUTPUT:

```
THE FACTORIAL OF A GIVEN NUMBER IS..120
```

Record at least 2 results

Assessment	
Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

7 b) Write a program to find GCD of two integers using non-recursive function and recursive function.

Aim: To find the GCD of two given integers by using the non recursive function

Description:

GCD means Greatest Common Divisor. i.e the highest number which divides the given number

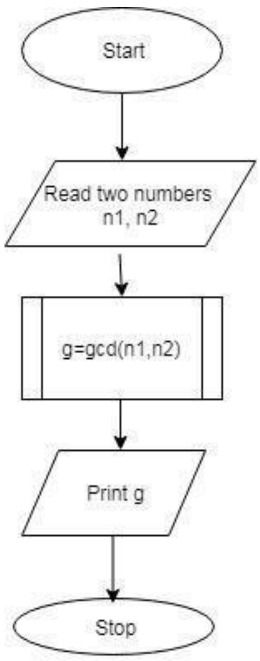
Ex: GCD(12,24) is 12

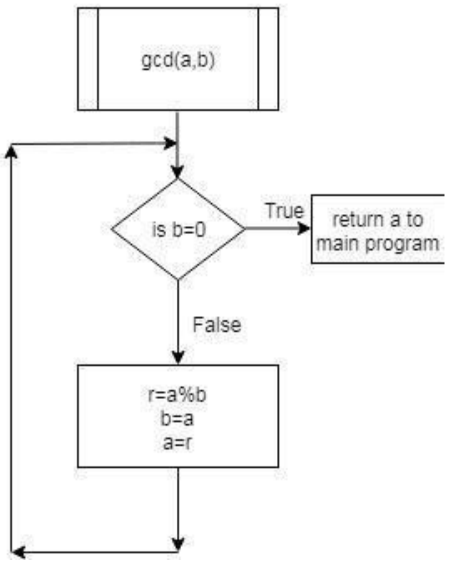
Formula: GCD= product of numbers/ LCM of numbers

Algorithm:

```
Step 1: start
Step 2: read a,b
Step 3: call sub program g=GCD(a,b)
Step 4: print the g value
Step 5: stop
Sub program:
    Step 1: if b=0 return a to main program
    Step 2: remainder=a%b
    Step 3:  a=b, b=remainder goto Step 1
```

Flowchart





Program:

```
#include<stdio.h>
int gcd(int a,int b);
int main()
{
int a,b;
int r,t;
        printf("Enter any two integers");
        scanf("%d%d",&a,&b);
        r=gcd(a,b);
        printf("GCD=%d",r);
        return 0;
}
int gcd(int a,int b)
{
int t,rem;
        while(1)
        {
            if(b>a)
            {
                  t=a;
                  a=b;
                  b=t;
            }
            if(b==0)
                 return a;
            else
            {
                 rem=a%b;
                 a=rem;
            }
        }
}
```

SAMPLE INPUT:

```
enter the two numbers whose gcd is to be found:5,25
```

EXPECTED OUTPUT:

```
GCD of  a,b is : 5
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Aim: To find the Gcd of two given integers by using the recursive function

Description: The greatest common divisor (gcd) of two or more integers, when at least one of them is not zero, is the largest positive integer that divides thenumbers without a remainder.

For example, the GCD of 8 and 12 is 4.

Algorithm:

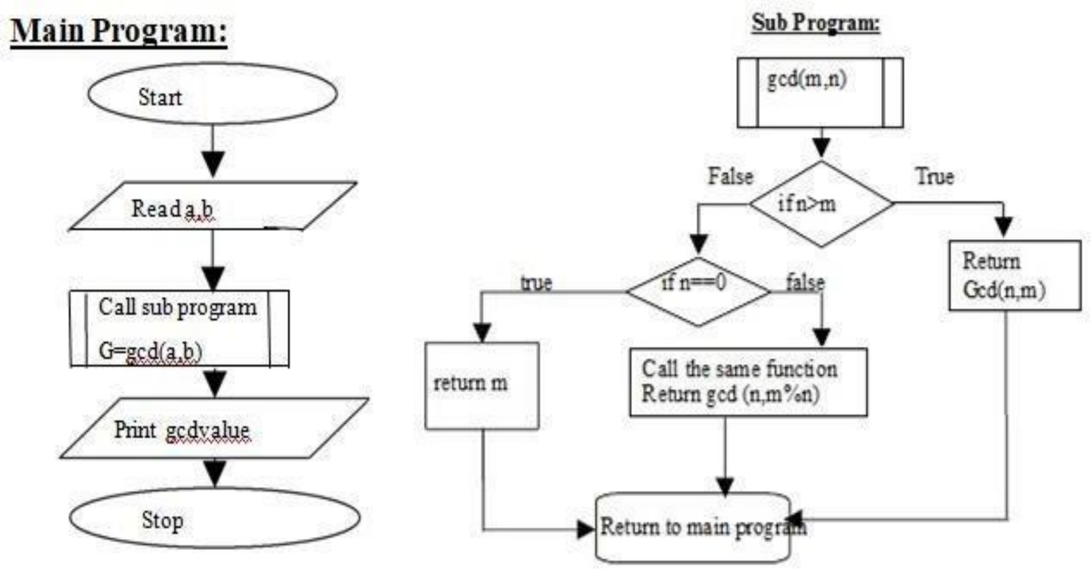
Main program:

```
Step 1: start
Step 2: read a,b
Step 3: call the sub program GCD(a,b) for print the value
Step 4: stop
```

Sub program: GCD(n,m)

```
    Step 1: if n>m return GCD(n,m)
    Step 2: if n==0 return m else goto step 3
    Step 3: return GCD (n,m%n)
  Step 4: return to main program
```

Flow CHART:



Program:

```
#include<stdio.h>
int gcdrecursive(int m,int n)
{        if(n>m)
                  return gcdrecursive(n,m);
           if(n==0)
                  return m;
           else
```

```
                    return gcdrecursive(n,m%n);         // return to the main program
}
int main()
{
        int a,b;
         printf("enter the two numbers whose gcd is to be found:");
         scanf("%d%d",&a,&b);
         printf("GCD of a,b is %d",gcdrecursive(a,b)); // return to the sub program
        return 0;
}
```

SAMPLE INPUT:

```
Enter the two numbers whose gcd is to be found: 5 25
```

EXPECTED OUTPUT:

```
GCD of a,b is : 5
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise:

1.write a program to multiply two numbers using recursion.
2.write a program to print Fibonacci numbers using recursion.
3.Write a program to find sum of natural numbers using recursion.
4.Write a program to calculate length of the string using recursion.
5. Write a program to solve Towers of Hanoi problem.
6. Write a program to add digits of a number using recursion.

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 8

8a) Write a C program using user defined functions to determine whether the given string is palindrome or not.

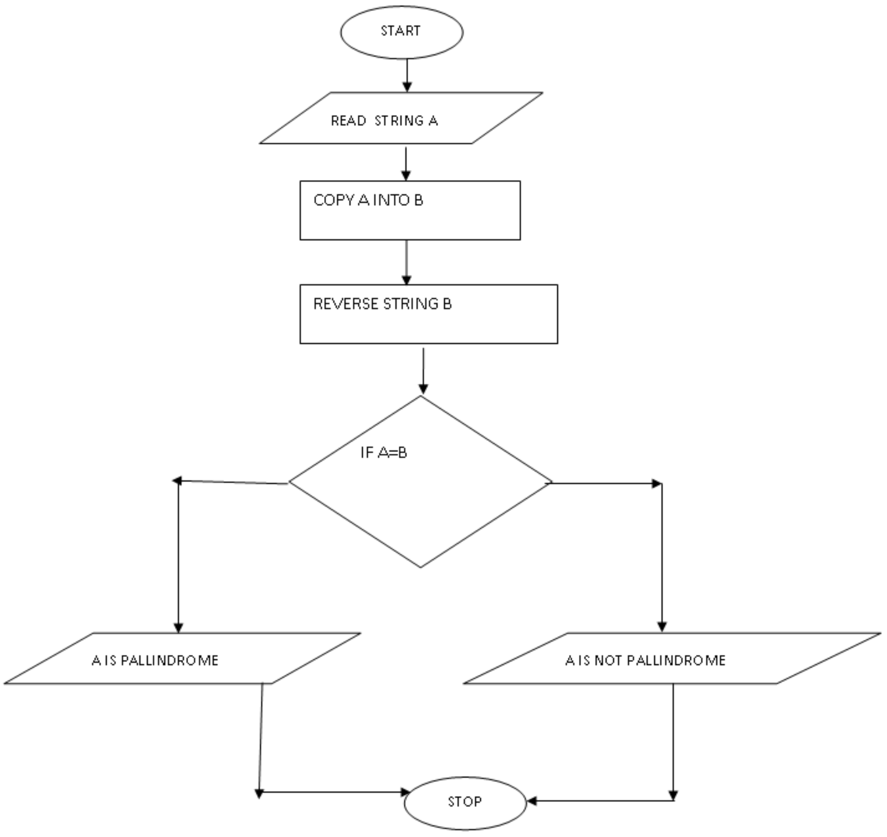
Aim: To determine if the given string is palindrome or not.

Description : Palindrome means string on reversal should be same as original
Ex: madam on reversal is also madam

Algorithm:

```
 Step 1: start
 Step 2: read string A
 Step 3: copy string A into B
 Step 4: reverse string B
 Step 5: compare A &B
         If A equals B to got step 6
               else goto step 7
Step 6:print given string A is palindrome
Step 7:print given string is not palindrome
Step 8: stop
```

Flow Chart:



Program:

```
#include <stdio.h>
#include <string.h>
int main()
{
  char string[25], reverse_string[25] = {'\0'};
   int i, length = 0, flag = 0;
```

```
printf("Enter a string \n");
   gets(string);
   for (i = 0; string[i] != '\0'; i++)
{
     length++;
}
  printf("The length of the string '%s' = %d\n", string, length);
   for (j=0,i = length - 1; i >= 0 ; i--,j++)
  {
     reverse_string[j] = string[i];
  }
  for (flag = 1, i = 0; i < length ; i++)
   {
     if (reverse_string[i] != string[i])
     {
                     flag = 0;
                     break;
  }
   if (flag == 1)
                   printf ("%s is a palindrome \n", string);
  else
                  printf("%s is not a palindrome \n", string);
return 0;
}
```

SAMPLE INPUT:

```
Enter a string
 madam
```

EXPECTED OUTPUT:

```
The length of the string 'madam' = 5
madam is a palindrome
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

8) b) Write a C Program to swap the values of two variables using i) Call by Value ii) Call by Reference

Aim: To Write a C Program to swap the values of two variables using
i) Call by Value ii) Call by Reference

Algorithm:

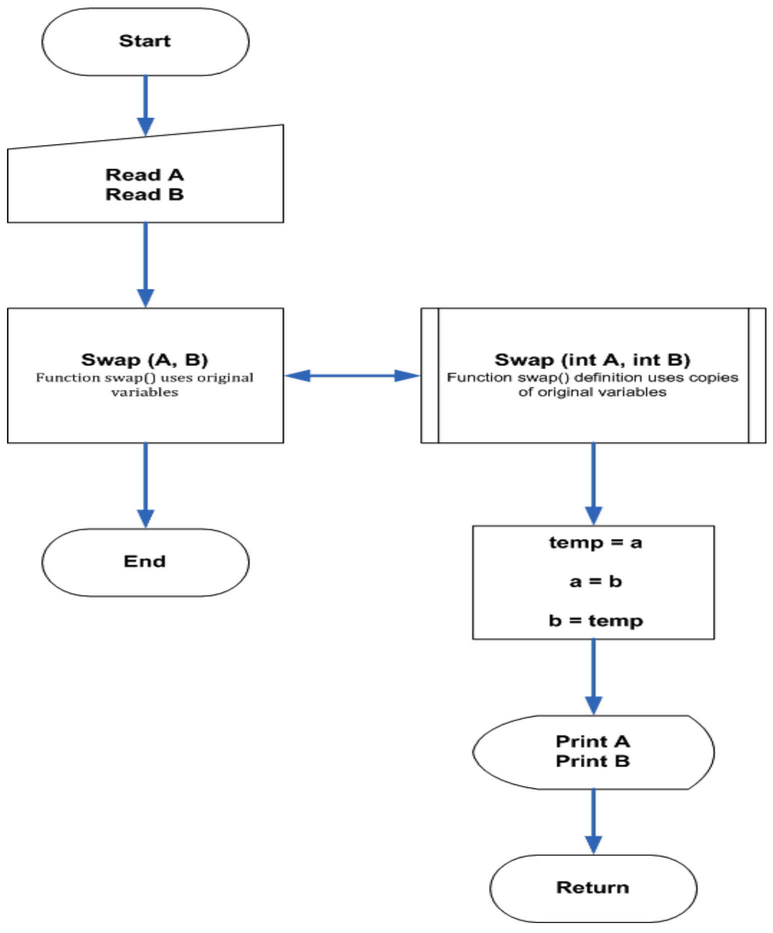
Main Program:

```
Step 1: start
Step 2: read a,b
Step 3: c a l l s w a p ( a , b )
Step 4: Stop
```

Sub Program:

```
Step 5: t=a;
Step 6:  a =b;
Step 7: b=t;
Step 8: print a ,b values
Step 9: return to main program
```

Flow chart:


Start
Read A
Read B
Swap (A, B)
Function swap() uses original variables
Swap (int A, int B)
Function swap() definition uses copies of original variables
End
temp = a
a = b
b = temp
Print A
Print B
Return

Program: write a program to swap using call by value

```
#include<stdio.h>
void swap(int , int);                  // Declaration of function
int main( )
{    int a,b;
          printf("Enter any two integers:");
            // call by value
             swap(a,b);                               // a and b are actual parameters
 }
void swap( int x, int y )                    // x and y are formal parameters
{    int t ;
     t = x ;
     x = y ;
    y = t ;
   printf ( "\nx = %d y = %d", x, y ) ;
}
```

SAMPLE INPUT:

```
          Enter any two integers: 10  20
```

EXPECTED OUTPUT:

```
          x=20 y=10
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

ii) Program to swap two number using call by reference.

```
#include <stdio.h>
void swap(int *a, int *b)
{   int temp;
            temp=*a;
            *a=*b;
            *b=temp;
}
int main()
{  int num1,num2;
          printf("Enter any Two Integers:");
           scanf("%d%d",&num1,&num2);
           swap(&num1,&num2);
           printf("Number1 = %d\n",num1);
           printf("Number2 = %d",num2);
  return 0;
}
```

SAMPLE INPUT:

```
          Enter any two integers: 2 3
```

EXPECTED OUTPUT:

```
Number1 = 3
Number2 = 2
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise:

1.Write a program which consists of user defined function to reverse a string

2.Design a user defined function to convert decimal number to binary.

3. Design a user defined function to print triangle as shown below

```
*
**
***
****
```

4. Design a user defined function to print triangle as shown below

```
1
1 2
1 2 3
1 2 3 4
```

5. Design a user defined function to generate first n prime numbers.

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 9

9 a)Write a program to find the sum of list of elements read from keyboard using dynamic memory allocation function malloc().

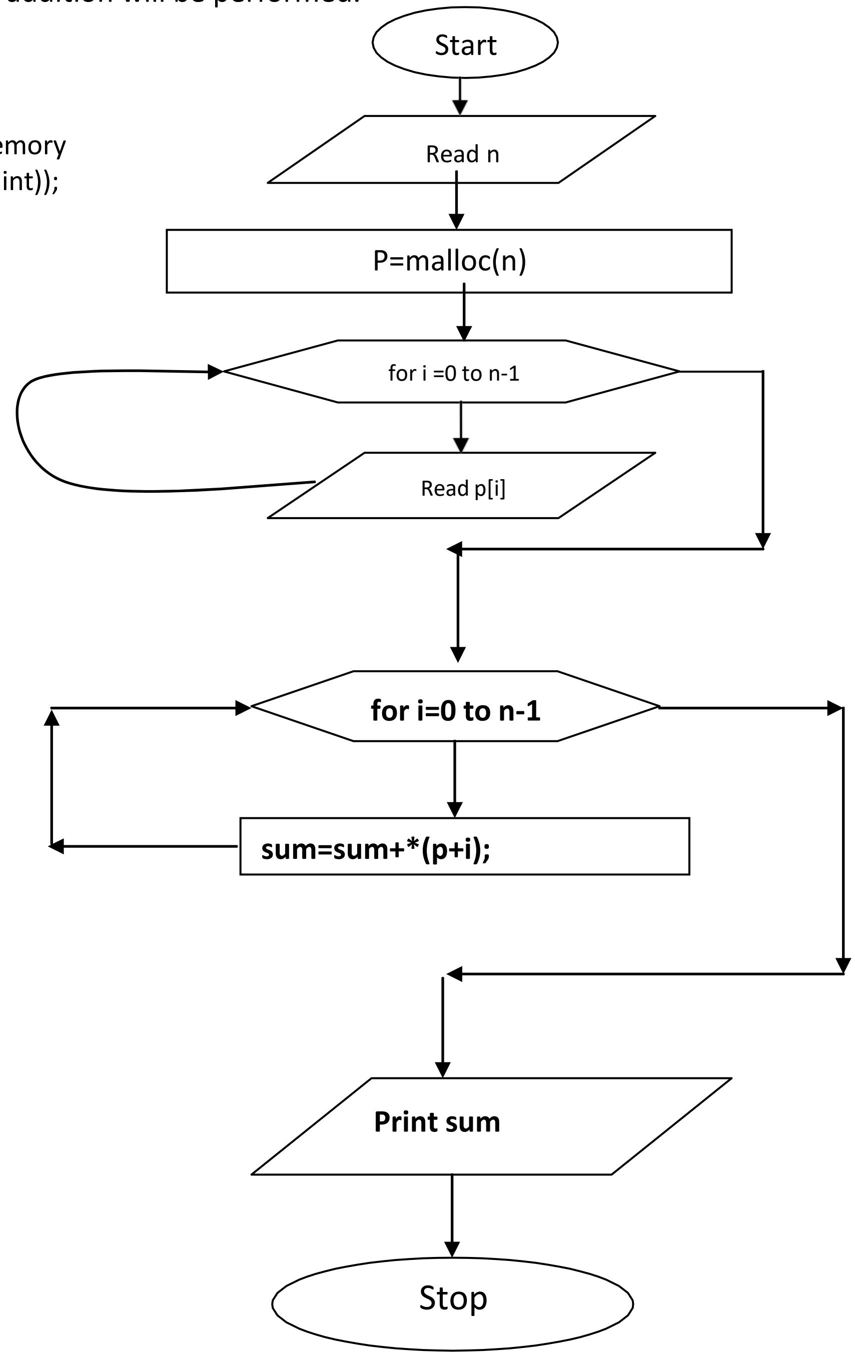
AIM: To find the sum of list of elements read from keyboard using dynamic memory allocation.

Description: Consider an integer array, by considering the name of the array as initial address of the first element, addition will be performed.

ALGORITHM:

```
Step 1:start
Step 2: read no. of elements N
Step 3:Dynamically allocate memory
        p=(int*)malloc(n*sizeof(int));
Step 4:read values into array
Step 6:sum=0
Step 3: for(i=0;i<N;i++)
Step 4: sum=sum+*(p+i)
step 5:print sum
step 6 stop
```

Flow Chart:



PROGRAM:

```
 #include<stdio.h>
#include<malloc.h>
int main( )
{
int *p,sum=0,i,n;
        printf("Enter How many elements to read:");
        scanf("%d",&n);
        p=(int*)malloc(n*sizeof(int));
        printf("enter elements into array");
        for(i=0;i<n;i++)
        scanf("%d",p+i);
        for(i=0;i<n;i++)
        sum=sum+*(p+i);
        printf("the addition of array elements is %d", sum);
return 0;
}
```

SAMPLE INPUT:

```
        Enter How many elements to read: 5
        enter elements into array 1 2 3 4 5
```

EXPECTED OUTPUT:

```
        the addition of array elements is 15
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

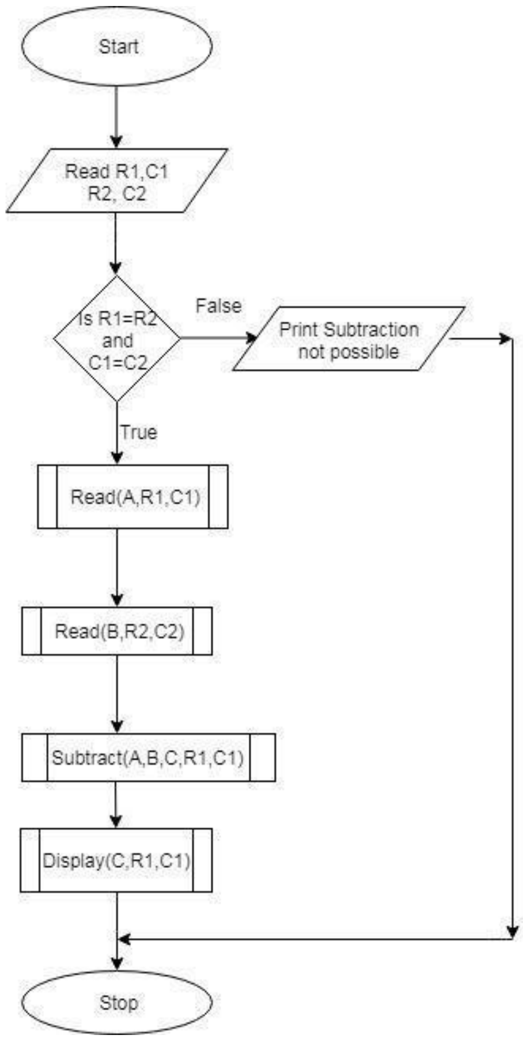
9 b)Write a program to perform subtraction of two matrices, Design functions to perform read ,display and subtract

AIM: To perform addition of two matrices.

ALGORITHM:

```
Step 1: Start
Step 2: Read row size and column size of two matrices
Step 3: if row size and column size of matrices are not equal goto Step 8
Step 4: call read function to read matrix A
Step 5: call read function to read matrix B
Step 6: call subtract function to perform subtraction
Step 7: call display function to print the resultant matrix
Step 8: Stop
```

Flow chart



PROGRAM:

```
#include<stdio.h>
void read_matrix(int a[5][5],int row,int col)
{
int i,j;
        for(i=0;i<row;i++)
                for(j=0;j<col;j++)
                scanf("%d",&a[i][j]);
}
void sub_matrix(int a[5][5],int b[5][5],int c[5][5],int row,int col)
{
int i,j;
        for(i=0;i<row;i++)
                for(j=0;j<col;j++)
                        c[i][j]=a[i][j]-b[i][j];
}
void display_matrix(int a[5][5],int row,int col)
{
 int i,j;
        for(i=0;i<row;i++)
        {
                for(j=0;j<col;j++)
                {
                        printf("%d\t",a[i][j]);
                }
                printf("\n");
        }
}
int  main( )
{
  int a[5][5],b[5][5],c[5][5], i,j,p,q,r,s;
  printf("ENTER ORDER OF A MATRIX\n");
  scanf("%d%d",&p,&q);
  printf("ENTER ORDER OF B MATRIX\n");
  scanf("%d%d",&r,&s);
  if(p==r&&q==s)
  {
        printf("ENTER A MATRIX\n");
        read_matrix(a,p,q);
        printf("ENTER B MATRIX\n");
        read_matrix(b,p,q);
        sub_matrix(a,b,c,p,q);
        printf(" After Subtraction :\n");
        display_matrix(c,p,q);
  }
  else
        printf("Subtraction not possible");
}
```

SAMPLE INPUT:

```
ENTER ORDER OF A MATRIX 2 2
ENTER ORDER OF B MATRIX 2 2
ENTER  A MATRIX
1  2
3  4
ENTER B MATRIX
0 1
2  3
```

EXPECTED OUTPUT:

```
 After Subtraction:
1      1
1      1
```

Record at least 2 results

Assessment

Not Satisfactory ☐
Needs Improvement ☐
Partially Executed ☐
Executed Successfully ☐

Signature of faculty with date

Exercise

1. Write a program to find the sum of list of elements read from keyboard using dynamic memory allocation function calloc().
2. Write a program to find the sum of list of elements read from keyboard using dynamic memory allocation function realloc().

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 10

10) a) Write a program to create book structure and display the contents of a book.

AIM: Write a program to create book structure and display the contents of a book.

Program:

```
#include<stdio.h>
struct  book
{
   char bname[50];
   int ssn;
   int pages;
  int rate;
};

int main()
{
struct book b1;
            printf("Enter Book SSN Number:");
            scanf("%d",&b1.ssn);
            printf("Enter Number of pages:");
            scanf("%d",&b1.pages);
            printf("Enter price:");
            scanf("%d",&b1.rate);
            fflush(stdin);
            printf("Enter Book Name:");
            gets(b1.bname);
            printf("\nName of the Book : %s\n ",b1.bname);
            printf("\nSSN of the Book : %d\n ",b1.ssn);
            printf("\nPages in the Book : %d\n ",b1.pages);
            printf("\nPrice of the  Book : %d\n",b1.rate);

   return(0);
}
```

SAMPLE INPUT:

```
            Enter Book SSN Number:123
            Enter Number of pages:200
            Enter price:100
            Enter Book Name:c programming
```

EXPECTED OUTPUT:

```
            Name of the Book : c programming
            SSN of the Book : 123
            Pages in the Book : 200
            Price of the  Book : 100
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

10) b) Write a C Program to Calculate Total and Percentage marks of a student using structure.

Program:

```
#include<stdio.h>
struct student
{
        int rl;
        char nm[20];
        int m1;
        int m2;
        int m3;
        int t;
        float per;
};
int main()
{
  struct student a;
         printf(" Enter RollNo, Name amd three sub marks\n");
         scanf("%d%s%d%d%d",&a.rl,&a.nm,&a.m1,&a.m2,&a.m3);
         a.t=a.m1+a.m2+a.m3;
         a.per=a.t/3.0;
         printf("rollno=%d\n",a.rl);
         printf("Name=%sk\n",a.nm);
         printf("m1=%d\n",a.m1);
         printf("m2=%d\n",a.m2);
         printf("m3=%d\n",a.m3);
         printf("total=%d\n",a.t);
         printf("per=%f\n",a.per);
  return 0;
}
```

SAMPLE INPUT:

```
Enter RollNo, Name and three sub marks
        12    rama         30 40  50
```

EXPECTED OUTPUT:

```
        rollno=12
        Name=rama
        m1=30
        m2=40
        m3=50
        total=120
        per=40.000000
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise

1. Write a program to read N Items rate and quantity and Calculate total billed amount.
2. Write a program for self referential structure.

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 11

11) a)Write a C program that uses functions to perform the following operations:
i) Reading a complex number ii) Writing a complex number
iii) Addition of two complex numbers iv) Multiplication of two complex

AIM: To perform arithmetic operations on complex numbers

Complex numbers of type a+ib
Addition: (a+ib)+(x+iy)=a+x+i(b+y)
Subtraction: (a+ib)-(x+iy)=a-x+i(b-y)
Multiplication: (a+ib)*(x+iy)= ax-by+i(ay+bx)
Division

(a+ib)/(x-iy) $=\frac{a+ib}{x+iy} * \frac{x-iy}{x-iy} = \frac{(a+ib)*(x-iy)}{x^2+y^2} = \frac{(ax+by)+i(bx-ay)}{x^2+y^2} = \frac{ax+by}{x^2+y^2} + i\,\frac{bx-ay}{x^2+y^2}$

ALGORITHM:

```
Step 1:start
Step 2: Read Two complex numbers c1, c2
Step 3: c3=c1+c2
Step 4: print c3
Step 5: c3=c1-c2
Step 6: print c3
Step 7: c3=c1*c2
Step 8: print c3
Step 9: c3=c1/c2
Step 10: print c3
Step 11: print c
Step 12: stop
```

PROGRAM:

```
#include<stdio.h>
#include<stdlib.h>
struct complex
{
        float real,img;
};

/*code for reading complex number*/
struct complex read_complex()
{ struct complex c;
        printf("enter real part of complex number");
        scanf("%f",&c.real);
        printf("enter Imaginary part of complex number");
        scanf("%f",&c.img);
        return c;
}

/*code for adding complex numbers*/
struct complex add_complex(struct  complex c1,struct complex c2)
{
struct complex c3;
        c3.real=c1.real+c2.real;
        c3.img=c1.img+c2.img;
        return c3;
}

/*code for subtraction of  complex numbers*/
struct complex sub_complex(struct complex c1,struct complex c2)
{
struct complex c3;
        c3.real=c1.real-c2.real;
        c3.img=c1.img-c2.img;
        return c3;
}

/*code for multiplication of  complex numbers*/
struct complex mul_complex(struct complex c1,struct complex c2)
{
struct complex c3;
        c3.real=c1.real*c2.real-c1.img*c2.img;
        c3.img= c1.img*c2.real+c2.img*c1.real;
        return c3;
}
```

```
/*code for division of  complex numbers*/
struct complex div_complex(struct complex c1,struct complex c2)
{
struct complex c3;
        c3.real=  (c1.real*c2.real+c1.img*c2.img)/(c2.real*c2.real+c2.img*c2.img);
        c3.img=  (c1.img*c2.real-c1.real*c2.img)/(c2.real*c2.real+c2.img*c2.img);
        return c3;
}
/*code for display of  complex number*/
void display_complex(struct complex c)
{
char sign;
        printf("The result is:");
        if(c.img<0)
        {
                sign='-';
                c. img=-c.img;
        }
        else
                sign='+';
        printf("%5f%ci%5f",c.real,sign,c.img);
}
int main()
{
int choice;
struct complex a,b,c;
        while(1)
        {
        printf("\n..........................................\n");
        printf("|Menu for operation complex numbers|\n ");
        printf("...........................................\n");
        printf("1.Addition \n ");
        printf("2.Subtraction \n ");
        printf("3.Multiplication \n ");
        printf("4.Division \n ");
        printf("5.Clear Screen \n ");
        printf("6.Exit Menu \n ");
        printf("Enter Your Choice: ");
        scanf("%d",&choice);
        switch(choice)
        {
        case 1:printf("You Have Selected Addition operation on complex Numbers\n");
                        printf("Enter First complex number\n");
                        a=read_complex();
                        printf("Enter Second complex Number\n");
                        b=read_complex();
                        c=add_complex(a,b);
```

```
                                        display_complex(c);
                                        break;
        case 2:printf("You Have Selected Subtraction operation on complex Numbers\n");
                                        printf("Enter First complex number\n");
                                        a=read_complex();
                                        printf("Enter Second complex Number\n");
                                        b=read_complex();
                                        c=sub_complex(a,b);
                                        display_complex(c);
                                        break;
       case 3:printf("You Have Selected Multiplication operation on complex Numbers\n");
                                        printf("Enter First complex number\n");
                                        a=read_complex();
                                        printf("Enter Second complex Number\n");
                                        b=read_complex();
                                        c=mul_complex(a,b);
                                        display_complex(c);
                                        break;

 case 4:printf("You Have Selected Division operation on complex Numbers\n");
                                        printf("Enter First complex number\n");
                                        a=read_complex();
                                        printf("Enter Second complex Number\n");
                                        b=read_complex();
                                        c=div_complex(a,b);
                                        display_complex(c);
                                        break;
     case 5: system(“cls”);
                                        break;
      case 6: exit(0);
                              default: printf("Invalid choice");
                              }
                    }
}
```

SAMPLE INPUT:

```
--------------------------------------
|Menu For Operation Complex Numbers|
 ---------------------------------------
 1.Addition
 2.Subtraction
 3.Multiplication
 4.Division
 5.Clear Screen
 6.Exit Menu
 Enter Your Choice:
Enter Your Choice: 1
You Have Selected Addition Operation On Complex Numbers
```

```
Enter First Complex Number
Enter Real Part Of Complex Number1
Enter Imaginary Part Of Complex Number2
Enter Second Complex Number
Enter Real Part Of Complex Number1
Enter Imaginary Part Of Complex Number2
```

EXPECTED OUTPUT:

```
THE RESULT IS:2.000000+I4.000000
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

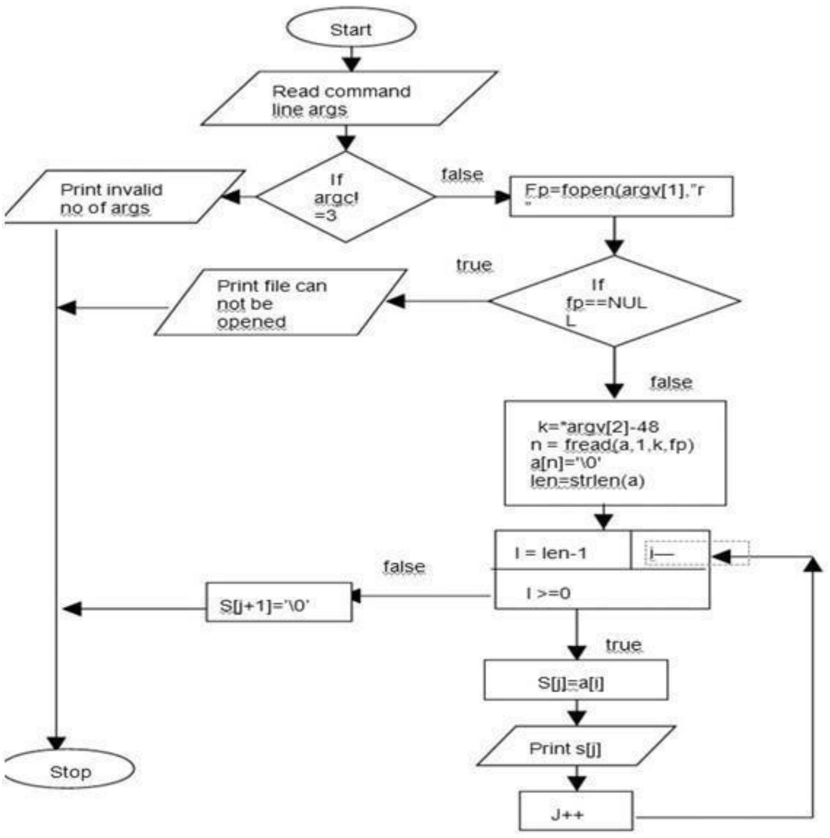
Signature of faculty with date

11b) Write a C program to reverse the first n characters in a file.
(Note: The file name and n are specified on the command line.)

Aim: To reverse the first n characters in a file

Algorithm:

```
Step 1: Start
Step 2: read the command line arguments
Step 3: check if arguments=3 or not
        If not print invalid no of arguments
Step 4: open source file in read mode
Step 5: if NULL pointer, then print file cannot be open
Step 6: Store no of chars to reverse in k
    K= *argv[2]-48
Step 7: read the item from file stream using fread
Step 8: Store chars from last position to initial position in another string(temp)
Step 9: print the temp string
Step 10: Stop
```



Program:

```
#include <stdio.h>
#include <string.h>
#include <process.h>
#include  <stdlib.h>
int  main(int argc, char *argv[])
{
  char a[15];
  char s[20];
  char n;
  int k;
  int j=0;
  int i;
  int len;
  FILE *fp;
  if(argc!=3)
  {
              puts("Improper number of arguments.");
               exit(0);
  }
  fp = fopen(argv[1],"r");
  if(fp == NULL)
  {
               puts("File cannot be opened.");
               exit(0);
  }
  k=atoi(argv[2]);
  n = fread(a,1,k,fp);
  a[n]='\0';
  len=strlen(a);
  for(i=len-1;i>=0;i--)
  {             s[j]=a[i];
                printf("%c",s[j]);
                j=j+1;
}
s[j+1]='\0';
return 0;
}
```

SAMPLE INPUT:

Input text file:

source.txt:

```
this is source
```

EXPECTED OUTPUT:

Command line arguments

```
C:\TURBOC~1\Disk\TurboC3\BIN>week11b source.txt 14
ecruos si siht
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise

1. Write a Program to Store Information of N Students like Name, Roll Number, marks and process result Using Structure
2. Write program to calculate difference between two time periods using structures.

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 12

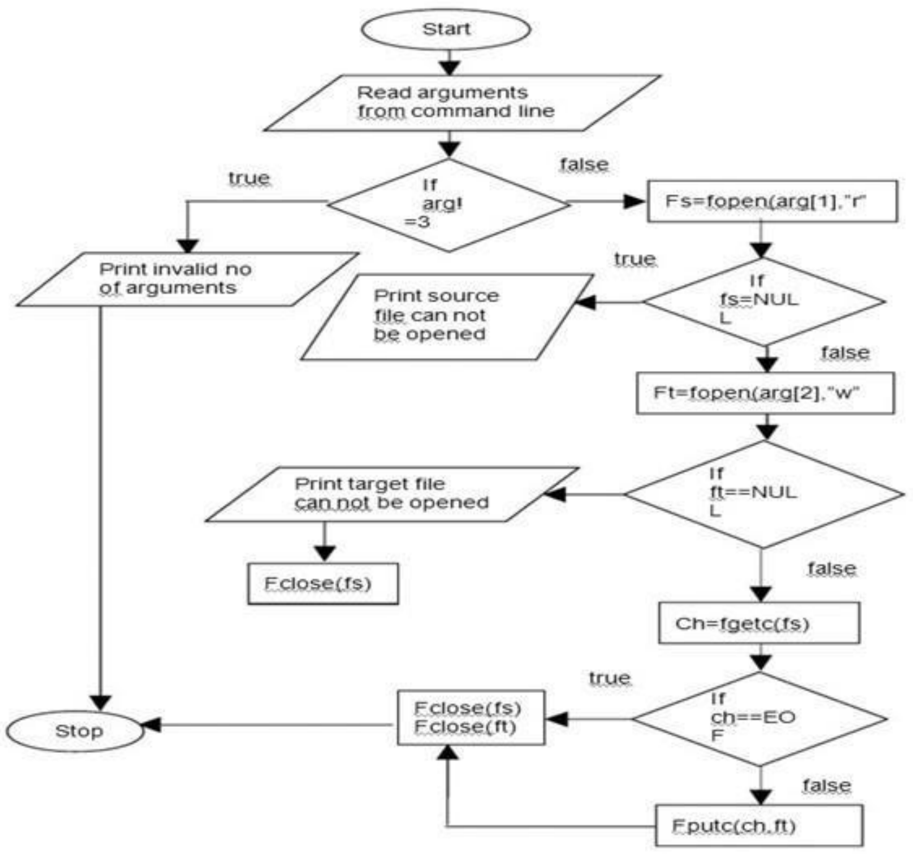
12)a) Write a C program to copy the contents of one file to another.

Aim: Program which copies one file to another

Algorithm:

```
Step 1: Start
Step 2: read file1,file2
Step 3: open source file in read mode
Step 4: if NULL pointer, then print source file can not be open
Step 5: open destination file in write mode
Step 6: if NULL pointer, then print destination file can not be open
Step 7: read a character from source file and write to destination file until EOF
Step 8 : Close source file and destination file
Step 9: Stop
```

Flow Chart:



Program:

```
#include<stdio.h>
#include<process.h>
int main()
{
FILE *ft,*fs;
int c=0;
        fs=fopen("a.txt","r");
        ft=fopen("b.txt","w");
        if(fs==NULL)
        {
                printf("Source file opening error\n");
                exit(1);
        }
        else
        if(ft==NULL)
        {
                printf("Target file opening error\n");
                exit(1);
        }
        while(!feof(fs))
        {
                fputc(fgetc(fs),ft);
                c++;
        }
        printf("%d bytes copied from 'a.txt' to 'b.txt'",c);
        c=fcloseall();
        printf("%d files closed",c);
        return 0;
}
```

SAMPLE INPUT:

```
        a.txt
        An array is a collection of elements of similar datatypes
```

EXPECTED OUTPUT:

```
        57 bytes copied from ‘a.txt’ to ‘b.txt’
        2 files closed
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

12 b) Write a program to merge two files into a third file.

Algorithm:

```
Step 1: Start
Step 2: read file1,file2,file3
Step 3: open file1 in read mode
Step 4: if NULL pointer, then print source file cannot be open
Step 5: open file3 in write mode
Step 6: if NULL pointer, then print file3 cannot be open
Step 7: read a character from file1 and write to file3 until EOF
Step 8 : Close file1
Step 9: open file2 in read mode
Step 10: if NULL pointer, then print source file can not be open
Step11: read a character from file2 and write to file3 until EOF
Step 12: Close file2 and file3
Step 13:Stop
```

Program :

```
#include<stdio.h>
int main()
{
        FILE *fp1,*fp2,*fp3;
        char file1[20],file2[20],file3[20],ch;
        puts("Program to merge two files. ..\n");
        puts("Enter first file name:");
        gets(file1);
        puts("Enter Second file name:");
        gets(file2);
        puts("Enter Destination file name:");
        gets(file3);
        fp1=fopen(file1,"r");
        fp2=fopen(file2,"r");
        fp3=fopen(file3,"w");
        if(fp1==NULL&&fp2==NULL)
                printf("Error opening file1 and file2.....\n");
        else
        {
                if(fp3==NULL)
                        printf("Error in creating destination file. ..\n");
                else
                {
                        while((ch=fgetc(fp1))!=EOF)
                        putc(ch,fp3);
                        while((ch=fgetc(fp2))!=EOF)
                        putc(ch,fp3);
                }
        printf("File Merging Sucessfull ...");
        fcloseall();
        }
        return 0;
}
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise

1. Write program to read name and marks of n number of students and store them in a file.
2. Write a program to read name and marks of n number of students from keyboard and store them in a file. If the file previously exits, add the information to the file.
3. Write a program to write all the members of an array of structures to a file using fwrite(). Read the array from the file and display on the screen.

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 13

13. Write a program for static implementation of stack

Algorithm:

INIT_STACK (STACK, TOP)

```
Algorithm to initialize a stack using array.
TOP points to the top-most element of stack.

1) TOP: = 0;
2) Exit
```

PUSH_STACK(STACK,TOP,MAX,ITEM)

```
Algorithm to push an item into stack.

1) IF TOP = MAX then
Print "Stack is full";
Exit;
2) Otherwise
TOP: = TOP + 1;       /*increment TOP*/
STACK (TOP):= ITEM;
3) End of IF
4) Exit
```

POP_STACK(STACK,TOP,ITEM)

```
Algorithm to pop an element from stack.

1) IF TOP = 0 then
    Print "Stack is empty";
    Exit;
2) Otherwise
    ITEM: =STACK (TOP);
    TOP:=TOP – 1;
3) End of IF
4) Exit
```

IS_FULL(STACK,TOP,MAX,STATUS)

```
Algorithm to check stack is full or not.
STATUS contains the result status.

1) IF TOP = MAX then
    STATUS:=true;
2) Otherwise
    STATUS:=false;
```

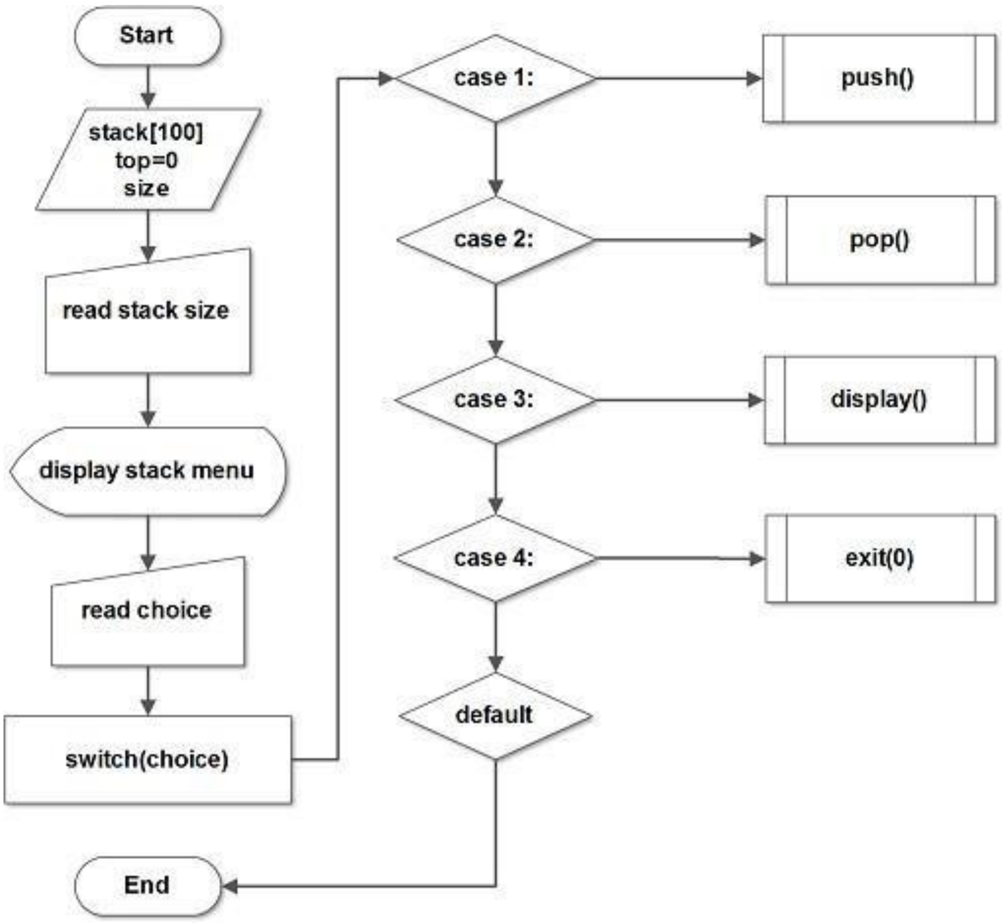
```
3)  End of IF
4)  Exit
```

IS_EMPTY(STACK,TOP,MAX,STATUS)

Algorithm to check stack is empty or not.
STATUS contains the result status.

```
1) IF TOP = 0 then
     STATUS:=true;
2) Otherwise
     STATUS:=false;
3)  End of IF
4)  Exit
```

Flow chart:



Program:

```
#include<stdio.h>
#include<process.h>
#define max 20
void push();
void pop();
void display();
int s[max],x,ch,top=-1;
int main()
{
        while(1)
        {
                printf("MENU OPTIONS\n");
                printf("1.PUSH\n");
                printf("2.POP\n");
                printf("3.DISPLAY\n");
                printf("4.EXIT\n");
                printf("ENTER YOUR CHOICE:");
                scanf("%d",&ch);
                switch(ch)
                {
                        case 1:push();
                            break;
                        case 2:pop();
                            break;
                        case 3:display();
                            break;
                        case 4:exit(0);
                            break;
                        default:printf("INVALID CHOICE\n");
                }
        }
}
void push()
{
        if(top>max-1)
                printf("STACK IS FULL\n");
        else
        {
                printf("ENTER ELEMENT TO BE INSERTED INTO THE STACK:");
                scanf("%d",&x);
                top++;
                s[top]=x;
        }
}
```

```
void pop()
{
        if(top<0)
                printf("STACK IS EMPTY\n");
        else
        {
                x=s[top];
                top--;
                printf("THE DELETED ELEMENT IS..%d\n",x);
        }
}
void display()
{
int i,count=0;
        if(top<0)
                printf("STACK IS EMPTY\n");
        else
         {
                printf("THE ELEMENTS IN THE STACK ARE..\n");
                for(i=top;i>=0;i--)
                printf("|%d|\n",s[i]);
        }
}
```

Sample input and Output

```
MENU OPTIONS
1.PUSH
2. POP
3. DISPLAY
4. EXIT
ENTER YOUR CHOICE :1
ENTER ELEMENT TO BE INSERTED INTO THE STACK :3
MENU OPTIONS
1. PUSH
2. POP
3. DISPLAY
4. EXIT
ENTER YOUR CHOICE:1
ENTER ELEMENT TO BE INSERTED INTO THE STACK:4
MENU OPTIONS
1. PUSH
2. POP
3. DISPLAY
4. EXIT
ENTER YOUR CHOICE :1
ENTER ELEMENT TO BE INSERTED INTO THE STACK:7
MENU OPTIONS
1. PUSH
2. POP
```

```
3. DISPLAY
4. EXIT
ENTER YOUR CHOICE:3
THE ELEMENTS IN THE STACK ARE..
|7|
|4|
|3|
MENU OPTIONS
1.PUSH
2. POP
3. DISPLAY
4. EXIT
ENTER YOUR CHOICE :2
THE DELETED ELEMENT IS..7
MENU OPTIONS
1. PUSH
2. POP
3. DISPLAY
4. EXIT
ENTER YOUR CHOICE:3
THE ELEMENTS IN THE STACK ARE..
|4|
|3|
MENU OPTIONS
1.PUSH
2. POP
3. DISPLAY
4. EXIT
ENTER YOUR CHOICE :4
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

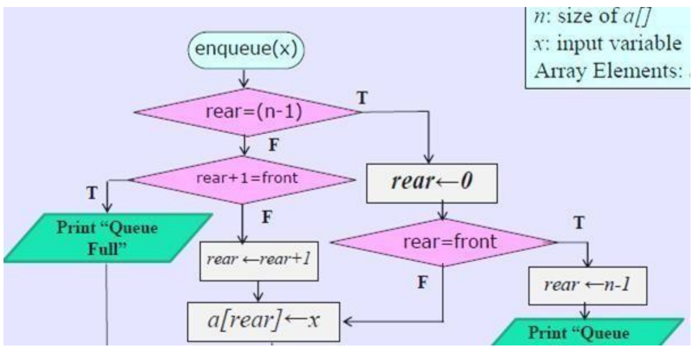
Signature of faculty with date

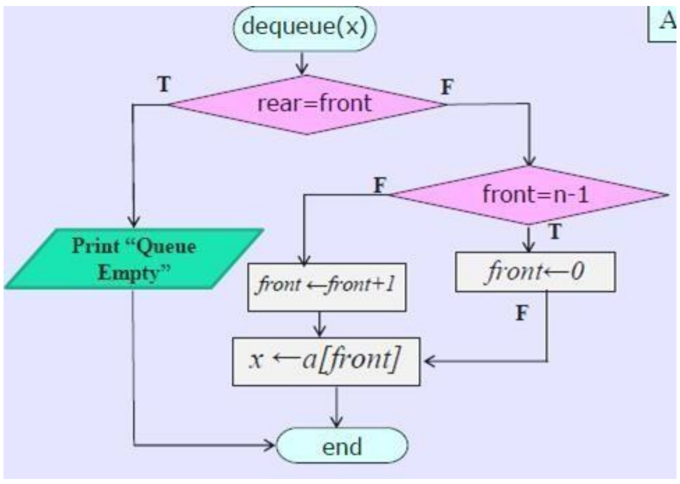
13. b) Write a C program for static implementation of Queue

Algorithm:

```
Step 1: IF REAR = MAX - 1
Write OVERFLOW
Go to step
[END OF IF]
Step 2: IF FRONT = -1 and REAR = -1
SET FRONT = REAR = 0
ELSE
SET REAR = REAR + 1
[END OF IF]
Step 3: Set QUEUE[REAR] = NUM
Step 4: EXIT
```

Flow Chart:





Program:

```
#include<stdio.h>
#include<stdlib.h>
#include<process.h>
#define max 20
void enqueue ();
void dequeue ();
void display();
int q[max],x,ch,rear=-1,front=-1;
int  main()
{         while(1)
          {
                    printf("\n***MENU FOR OPERATIONS ON QUEUE***\n");
                    printf("1.INSERT\n");
                    printf("2.DELETE\n");
                    printf("3.DISPLAY\n");
                    printf("4.CLEAR\n");
                    printf("5.EXIT\n");
                    printf("ENTER YOUR CHOICE:");
                    scanf("%d",&ch);
                    switch(ch)
                    {
                    case 1: enqueue ();
                         break;

                    case 2: dequeue ();
                         break;

                    case 3:display();
                         break;

                    case 4:system("cls");
                                        break;

                    case 5:exit(0);
                           break;
                    default:printf("INVALID CHOICE\n");
                    }
          }
return 0;
}
void enqueue()
```

```
{
        if(rear==max-1)
                printf("QUEUE OVERFLOW\n");
        else
        {
                printf("ENTER ELEMENT TO BE INSERTED INTO THE QUEUE:");
                scanf("%d",&x);
                rear++;
                q[rear]=x;
        }
}

void dequeue ()
{
        if(front==rear)
                printf("QUEUE UNDERFLOW\n");
        else
        {
                front++;
                x=q[front];
                printf("%d IS DELETED\n",x);
        }
}
void display()
{
int i;
        printf("THE ELEMENTS IN THE QUEUE ARE..\n");
        for(i=front+1;i<=rear;i++)
                printf("%d-",q[i]);
}
```

Sample Input and Expected Output:

```
***MENU FOR OPERATIONS ON QUEUE***
1. INSERT
2. DELETE
3. DISPLAY
4. EXIT
ENTER YOUR CHOICE:1
ENTER ELEMENT TO BE INSERTED INTO THE QUEUE:3

*** MENU FOR OPERATIONS ON QUEUE ***
1.INSERT
2. DELETE
3. DISPLAY
4. EXIT
```

```
ENTER YOUR CHOICE:1
ENTER ELEMENT TO BE INSERTED INTO THE QUEUE 4
*** MENU FOR OPERATIONS ON QUEUE***
1. INSERT
2. DELETE
3. DISPLAY
4. EXIT
ENTER YOUR CHOICE 1

ENTER ELEMENT TO BE INSERTED INTO THE QUEUE 7

*** MENU FOR OPERATIONS ON QUEUE***
1. INSERT
2. DELETE
3. DISPLAY
4. EXIT
ENTER YOUR CHOICE 3

THE ELEMENTS IN THE QUEUE ARE..
3-4-7-

*** MENU FOR OPERATIONS ON QUEUE***
1. INSERT
2. DELETE
3. DISPLAY
4. EXIT
ENTER YOUR CHOICE 2

3 is DELETED

 *** MENU FOR OPERATIONS ON QUEUE***
1. INSERT
2. DELETE
3. DISPLAY
4. EXIT
ENTER YOUR CHOICE 3

THE ELEMENTS IN THE QUEUE ARE..
4-7-
```

Record at least 2 results

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise

1. Write a program for matching parenthesis using stack
2. Write a program for circular queue

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 14

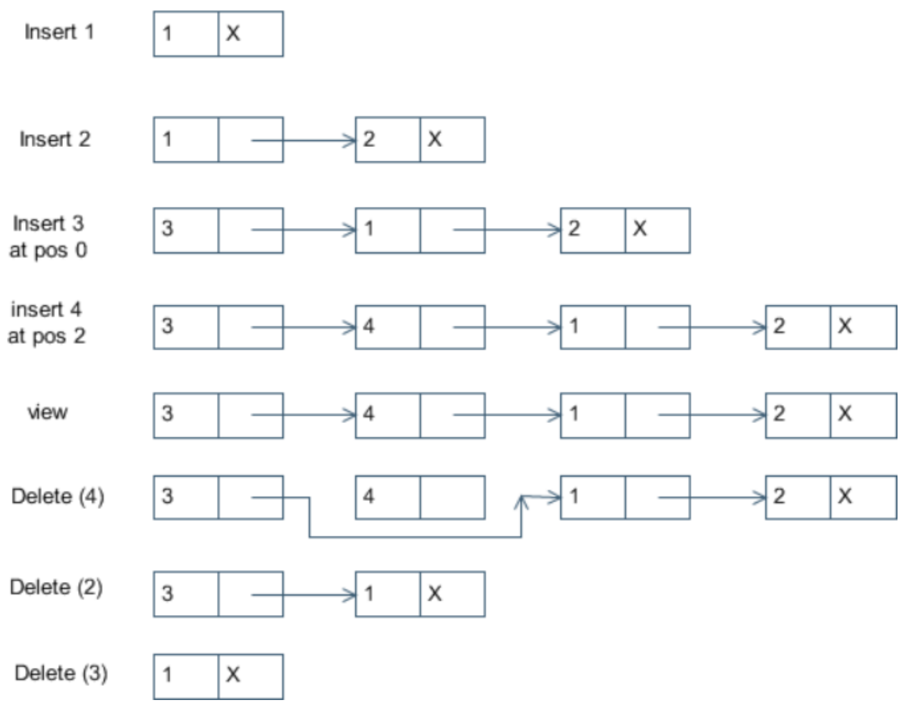
14. Write a C program that uses functions to perform various operations on singly linked list.

AIM: To perform various operations on singly linked list.

Description:

In this program we have to create a single linked list, insert the elements into that list ,delete the some elements from that list and then perform the sorting operation and traversal operation on that created linked list

Various operation on single linked list



Algorithm :

```
 Step 1: Start
Step 2: Declare a structure named linked-list
Step 3: Declare the pointers next, first, fresh, ptr
Step 4: Print main menu
Step 5: Read choice
Step 6: Switch(choice)
Step 7: If(choice==1)
```

```
            Assign fresh=malloc(size of (node))
            Read the element fresh->data
Read the choice where to insert
7.4:Switch(choice)
7.4.1: If choice==1
7..4.2: Call the function IBegin()
          7.4.3: If choice==2
7.4.4: Call the function Iend()
7.4.5: If choice==3
7.4.6: Call the function Imiddle()
Step 8: If(choice==2)
8.1: Read the position to delete
8.2: Switch(choice)
8.2.1: If choice==1
8..2.2: Call the function DBegin()
          8.2.3: If choice==2
8.2.4: Call the function Dend()
8.2.5: If choice==3
8.2.6: Call the function Dmiddle()
Step 9: If choice==3
9.1 Call function view
114Step 10: If choice==4
10.1 Exit()
Step 11: Start insert function
Step 12: If(first==null)
Step 13: First->data=e
Step 14: First->next=null
Step 15: Else declare new node
Step 16:fresh->data=e
Step 17: If choice=1
Step 18: frsh->next=first
Step 19: first=fresh
Step 20:if choice=2
Step 21: ptr=first
Step 22: ptr->next=fresh
Step 23: fresh->next=full
Step 24: If choice =3
Step 25: Enter the position
Step 26:at p-1 node
Step 27: fresh->next= ptr->next
Step 28: ptr->next=fresh
Step 29: for delete function
Step 30: If first!=null
Step 31: Enter the position to delete
Step 32: If choice=1
115Step 33: d=first->data
Step 34: first=first->next
Step 35: if choice=2
Step 36: ptr=first
Step 37: Traverse to last node
Step 38: d=ptr->next->data
Step 39: ptr ->next=ptr->next->next
Step 40: Print d value
Step 41: for function view
Step 42: for ptr=first and ptr!=null and ptr=ptr->next
Step 43: Print ptr->data
Step 44: End
```

PROGOGRAM:

```
#include<stdio.h>
#include<stdlib.h>
struct lnode
{
        int data;
        struct lnode *next;
};
typedef struct lnode * lptr;

/*function to create a node*/
lptr create_node()
{
lptr node;
node=(lptr)malloc(sizeof(struct lnode));
return node;
}

/*function to insert an element at end of list*/
lptr insert_end(lptr head,int x)
{
lptr p,temp;
        p=create_node();
        p->data=x;
        p->next=NULL;
        if(head==NULL)
        return p;
        else
        {
        temp=head;
        while(temp->next!=NULL)
        temp=temp->next;
        temp->next=p;
        return head;
        }
}
```

```
/*function to insert an element at front of existing list*/
lptr insert_front(lptr head,int x)
{
lptr p,temp;
        p=create_node();
        p->data=x;
        p->next=NULL;
        if(head==NULL)
                return p;
        else
        {
                p->next=head;
                head=p;
                return head;
        }
}

/*function to insert element after a node*/
lptr insert_after(lptr head,int x,int pos)
{
lptr p,temp,q;
int c=1;
        p=create_node();
        p->data=x;
        p->next=NULL;
        if(head==NULL)
        return p;
        else
        {
        temp=head;
        while(c<=pos)
        {
        q=temp;
        temp=temp->next;
        c++;
        }
        q->next=p;
        p->next=temp;
        return head;
        }
}

/* function to insert a node before a selected node*/
lptr insert_before(lptr head,int x,int pos)
{
lptr p,temp,q;
int c=1;
        p=create_node();
```

```
            p->data=x;
            p->next=NULL;
            if(head==NULL)
                        return p;
            else
            {           temp=head;
                        while(c<=(pos-1))
                        {
                                    q=temp;
                                    temp=temp->next;
                                    c++;
                        }
                        if(c==1)
                        {
                                    p- >next=head;
                                    head=p;
                        }
                        else
                        {
                                    q- >next=p;
                                    p->next=temp;
                        }
            return head;
            }
 }

/*function to insert a node at end of list*/
lptr delete_end(lptr head)
{
lptr temp,prev;
            temp=head;
            if(head->next==NULL)
            {
            free(temp);
            head=NULL;
            return head;
            }
            else
            {  while(temp->next!=NULL)
            {
            prev=temp;
            temp=temp->next;
            }
            prev->next=NULL;
            printf("\n %d is deleted from list",temp->data);
            free(temp);
            return head;
            }
}
```

```
/*function to delete a node from front of list*/
lptr delete_front(lptr head)
{
lptr temp;
            temp=head;
            head=head->next;
            printf ("\n%d is deleted",temp->data);
            free(temp);
            return(head);
}

/*function to display data present in all nodes in a linked list*/
void display(lptr head)
{
lptr temp;
            temp=head;
            printf("\n\n");
            while(temp!=NULL)
            {
            printf("%d->",temp->data);
            temp=temp->next ;
            }
}

/*function to print number of nodes present in a linked list*/
int node_count(lptr head)
{
lptr temp;
            int ncount=0;
            temp=head;
            while(temp!=NULL)
            {
            temp=temp->next;
            ncount++;
            }
            return ncount;
}



int  main()
{
lptr head=NULL;
int choice,item,pos;
while(1)
{
printf("enter choice \n1.insert end \n2.insert front \n3.insert after\n4.insert before\n");
printf("5.del end\n6.del front\n7.dislay\n8.node count \n9.clear screen\n10.exit");
```

```
scanf("%d",&choice);
switch(choice)
{
case 1 : printf("enter value into new node");
         scanf("%d",&item);
         head=insert_end(head,item);
         printf("\n\nelement inserted at end");
         break;
case 2 : printf("\n\nenter value into new node");
         scanf("%d",&item);
         head=insert_front(head,item);
         printf("\n\nelement inserted at end");
         break;
case 3 : printf("\nenter value into new node");
         scanf("%d",&item);
         printf("\n\nenter position from 1 to %d",node_count(head));
         scanf("%d",&pos);
         head=insert_after(head,item,pos);
         printf("\n\nelement inserted after %d pos",pos);
         break;
case 4 : printf("enter value into new node");
         scanf("%d",&item);
         printf("\nenter position from 1 to %d",node_count(head));
         scanf("%d",&pos);
         head=insert_before(head,item,pos);
         printf("\nelement inserted before %d pos",pos);
         break;
case 5 : if(head==NULL)
                 printf("empty list");
         else
                 head=delete_end(head);
         break;
case 6 : if(head==NULL)
                 printf("empty list");
         else
                 head=delete_front(head);
         break;
case 7 : if(head==NULL)
                 printf("empty list");
         else
                 display(head);
         break;
case 8 : if(head==NULL)
                 printf("empty list");
         else
                 printf("\nno of nodes are %d",node_count(head));
         break;
case 9 : system(“cls”);
         break;
case 10 : exit(0);
default : printf("invalid choice enter choice again ");
break;
}
}
}
```

Sample input & Output:

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise

1. Write a program to print the given list in reverse order
2. Write a program to merge two lists
3. Write a program to split given list into two based on odd and even places

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Week 15

15 a).Write a program that implements stack (its operations) using a singly linked list

DESCRIPTION:

In this program we have to implement the stack operation by using the arrays. Here they stack operation are push and pop. Push operation is used to insert the elements into a stack and pop operation is used to remove the elements in to a stack

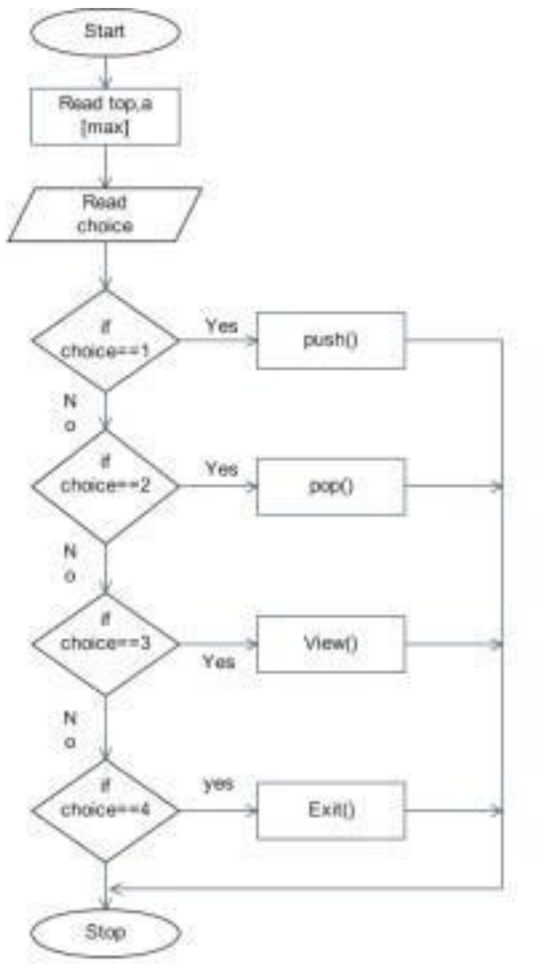
ALGORITHM:

ALGORITHM FOR PUSH operation

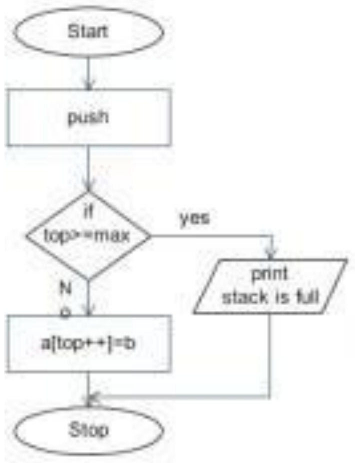
```
Function Push(s,top,x)
Step 1: [Check for stack overflow]
        If top>=n
        Then printf("stack overflow")
         Return
Step 2: [Increment Top]
        Top<-top+1
Step 3: [ Insert element]
        S[top]<-x
        Step 4:[finished]
        Return
```

ALGORITHM FOR POP OPERATION

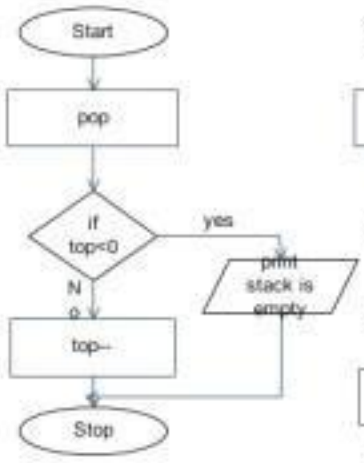
```
Function POP(s,top)
Step 1: [Check for stack underflow]
        If top=0
        Then printf("stack underflow")
         Exit
Step 2: [Decrement Top]
        Top<-top-1
Step 3: [Return former top element of stack]
Return(S[top+1])
Step 4:[finished]
Return
```



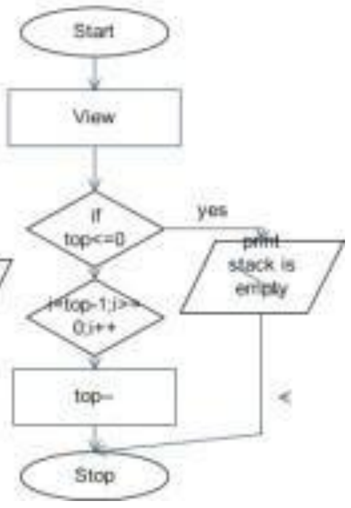
push()



pop()



view()



PROGRAM:

```
#include<stdio.h>
#include<stdlib.h>
struct stack
{
        int data;
        struct stack *next;
};
typedef struct stack * sptr;
sptr createnode()
{
sptr node;
        node=(sptr)malloc(sizeof(struct stack));
        return node;
}
sptr push(sptr head,int x)
{
sptr p,temp;
        p=createnode();
        p->data=x;
        p->next=NULL;
        if(head==NULL)
                return p;
        else
        {
                p->next=head;
                head=p;
                return head;
        }
}
sptr pop(sptr head)
{
sptr temp;
        temp=head;
        head=head->next;
        printf("%d is deleted",temp->data);
        free(temp);
        return(head);
}
void display(sptr head)
{
sptr temp;
        temp=head;
        while(temp!=NULL)
        {
        printf("%d\n",temp->data);
        temp=temp->next;
```

```
            }
}
int main()
{
 sptr top=NULL;
 int choice,item;
            while(1)
            {
            printf("\n----------------------------------------\n");
            printf("| MENU FOR OPERATIONS ON STACK |");
            printf("\n----------------------------------------\n");
            printf("1.push\n2.pop\n3.display\n4.exit");
            printf("\n---------------------------------------\n");
            printf("Enter Your choice:");
            scanf("%d",&choice);
            switch(choice)
            {
            case 1: printf("\n enter element to push");
                        scanf("%d",&item);
                        top=push(top,item);
                        printf("element %d is pushed succesfully" ,item);
                        break;
            case 2: if(top==NULL)
                                    printf("stack is empty");
                        else
                                    top=pop(top);
                        break;
            case 3: if(top==NULL)
                        printf("stack is empty");
                      else
                        display(top);
                        break;
            case 4: exit(0);
            }
      }
}
```

Sample Input & Output:

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

15 b) Write a C program that implements Queue (its operations) using a singly linked list.

Description:

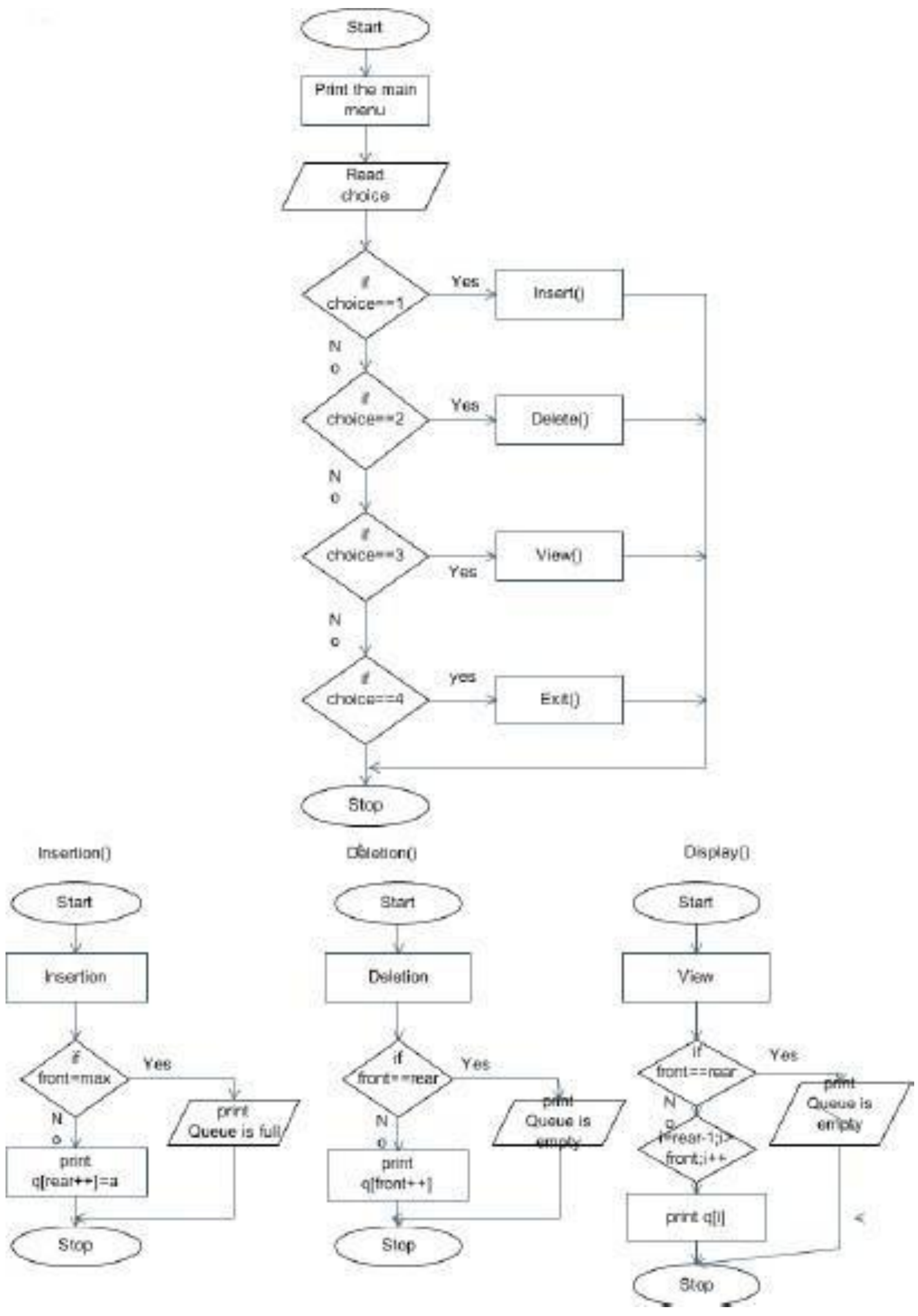
In this program we have to implement the Queue operation by using the arrays. Here they Queue operation are push and pop. Insert operation is used to insert the elements into a Queue and Delete operation is used to remove the elements in to a Queue.

ALGORITHM:

```
ALGORITHM FOR INSERTING AN ELEMENT IN TO A QUEUE:
Function QINSERET(Q,F,R,N,Y)
Step 1: [overflow]
        If R>=N
        Then printf(" overflow")
        Return
Step 2: [Increment rear pointer]
        R<-R+1
        Step 3: [ Insert element]
        Q[R]<-y
        Step 4: [Is front pointer properly set?]
        If F=0
        Then f<-1
        Return

ALGORITHM FOR DELETING AN ELEMENT FROM A STACK:
Function QDELETE(Q,F,R)
Step 1: [Underflow]
        If F=0
        Then printf("Queue underflow")
        Return(0)
Step 2: [Delete element]
        y<-q[f]
Step 3: [Is Queue Empty?]
        If F=R
        Then F=R=0
        Else
        F=F+1
Step 4:[Return element]
        Return(r)
```

FLOW CHART


Start
Print the main menu
Read choice
if choice==1
Yes
Insert()
No
if choice==2
Yes
Delete()
No
if choice==3
Yes
View()
No
if choice==4
yes
Exit()
Stop
Insertion()
Start
Insertion
if front=max
Yes
print Queue is full
No
print q[rear++]=a
Stop
Deletion()
Start
Deletion
if front==rear
Yes
print Queue is empty
No
print q[front++]
Stop
Display()
Start
View
if front==rear
Yes
print Queue is empty
No
i=rear-1;i>front;i++
print q[i]
Stop

PROGRAM:

```
#include<stdio.h>
#include<stdlib.h>
struct queue
{
        int data;
        struct queue *next;
};
typedef struct queue * qptr;

qptr createnode()
{
qptr node;
node=(qptr)malloc(sizeof(struct queue));
return node;
}

qptr insert_q(qptr r,qptr *f,int x)
{
qptr p,temp;
        p=createnode();
        p->data=x;
        p->next=NULL;
        if(*f==NULL)
        {       r=p;
                *f=r;
        }
        else
        {
        while(r->next!=NULL)
                r=r->next;
        r->next=p;
        }
return r;
}

void delete_q(qptr *front)
{
qptr temp;
        temp=*front;
        *front=(*front)->next;
        printf ("\n\n%d is deleted",temp->data);
        free(temp);
}
```

```
void display(qptr front)
{
qptr temp;
          temp=front;
          while(temp!=NULL)
          {
          printf("\t%d\n",temp->data);
          temp=temp->next;
          }
}
int main()
{qptr rear,front;
int choice,item;
          front=rear=NULL;
          while(1)
          {
                    printf("\n................................\n");
                    printf("|MENU FOR OPERATIONS ON QUEUE|");
                    printf("\n................................\n");
                    printf("1.INSERT \n 2.DELETE\n3.DISPLAY\n4.EXIT\n Enter choice:");
                    scanf("%d",&choice);
                    switch(choice)
                    {
                              case 1: printf("\n enter element to insert");
                                        scanf("%d",&item);
                                        rear=insert_q(rear,&front,item);
                                        printf("\n\nelement %d is inserted succesfully" ,item);
                                        break;
                              case 2: if(front==NULL)
                                        printf("\n\nqueue is empty");
                                        else
                                        delete_q(&front);
                                        break;
                              case 3: if(front==NULL)
                                                  printf("\n\nqueue is empty");
                                        else
                                                  display(front);
                                        break;
                              case 4: exit(0);
                              default:printf("Invalid choice...Try Again\n");
                              break;
                    }
          }
}
```

Sample Input and Expected Output:

Assessment

Not Satisfactory	☐
Needs Improvement	☐
Partially Executed	☐
Executed Successfully	☐

Signature of faculty with date

Exercise

1. Write a program to search a list for a given key value.
2. Write a program for implementation of circular queue using linked list.

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Case Studies

Case 1: Student Record Management System

Student Record System is used by education establishments to manage student data. Student Record System provides capabilities for entering student details and manage many other student-related data. In this Student Record System, user can create, display, search, modify and delete student record from a student file.

Algorithm:

```
Step1: Create a structure for the student with his Roll no., Name, Branch, cgpa in 8 subjects,
sgpa
Step2: Print "Student Record System"
        Print "1. Add Student"
        Print "2. Display Student Details"
        Print "3. Search for a Student"
        Print "4. Modify a Student Details"
        Print "5. Delete a Student"
        Print "6. Exit"
        Print "Enter your choice(1-6): "
Step3:  Accept the user's choice.
Step4:  If choice = 1, call  add() function to insert new student record in the file
        else if choice=2, call display() function to display the details of all the students
             in the  file
        else if choice=3, call search() function to search for a student given his Roll no.
        else if choice=4, call modify() function to update the details of a student with
             a given Roll no.
        else if choice=5, call delete() function to delete student details from the file
        else if choice=6, goto step6
Step5: Check if the user wants to perform any other operation. If so, goto Step 2.
Step6: Stop
```

Functions:

add()

```
Step1: Accept the details of the new student to be created into the structure variable
Step2: Open the student file in append mode
        if the file pointer contains NULL
             Print "File cannot be opened"
                  return to main program
Step3: write the structure variable at the end of the student file
Step4: Close the file
Step5: Return to main program
```

display()

```
Step1: Open the student file in read mode
            if the file pointer contains NULL
                    Print "File cannot be opened"
                            return to main program
Step2: read the next record from the file into structure variable
Step3: Print the student data
Step4: Repeat Step2 and Step3 until end of file is reached
Step5: Close the file
Step6: Return to main program
```

search()

```
Step1: Open the student file in read mode
            if the file pointer contains NULL
                    Print "File cannot be opened"
                    return to main program
Step2: Read the Roll no. of the student to be searched from the user
Step3: Read the next record from the student file into structure variable
Step4: Compare the Roll no. in the structure variable with the Roll no. read from the user
Step5: If both are matching, display other details of the student on the screen, goto Step8
Step6: If match is not found, Repeat Step3, Step4 and Step5 until end of file is reached
Step7: If end of file is reached, Print “Student record does not exist”.
Step8: Close the file
Step9: Return to main program
```

modify()

```
Step1: Open the student file in read write mode
            if the file pointer contains NULL
                    Print "File cannot be opened"
                            return to main program
Step2: Read the Roll no. of the student to be modified from the user
Step3: Read the next record from the student file into structure variable
Step4: Compare the Roll no. in the structure variable with the Roll no. read from the user
Step5: If match is found, read the other details of the student from the user

Step6: Change the position of the file pointer to point to the current record
Step7: Write the new structure variable to the file, goto Step 10
Step8: If match is not found, Repeat Step3, Step4 and Step5 until end of file is reached
Step9:  If end of file is reached, Print "Student Record not found"
Step10: Close the file
Step11: Return to main program
```

delete()

```
Step1: Open the student file in read mode
           if the file pointer contains NULL
                  Print "File cannot be opened"
                           return to main program
Step2: Create a new student file in write mode
Step2: Read the Roll no. of the student to be deleted from the user
Step3: Read the next record from the student file into structure variable
Step4: Compare the Roll no. in the structure variable with the Roll no. read from the user
Step5: If both are not matching, write the details in the structure variable into new student file,
           goto Step7
Step6: If match is found, skip this record.
           Repeat Step3, Step4 and Step5 until end of file is reached
Step7: Close the files
Step8: Remove the old student file, Rename new student file as student file
Step9: Return to main program
```

Sample Code:

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

void add();          //to add to  list
void del();          //to delete from list
void modify();     //to modify a record
void display();    //display whole list
void search();     //find a particular record

typedef struct
{
   char name[100];
   char branch[50];
   int roll;
   float sgpa[8];
   float cgpa;
}student;


int main()
{
```

```
int option;
char another;
while(1)
{
   printf("\n\n\t\t\tStudent Record System\n \n ");
   printf("\n\n\t\t\t1. ADD Student Record");
   printf("\n\n\t\t\t2. DISPLAY Student LIST ");
   printf("\n\n\t\t\t3. SEARCH Student Record ");
   printf("\n\n\t\t\t4. MODIFY Student Record");
   printf("\n\n\t\t\t5. DELETE Student Record");
   printf("\n\n\t\t\t0. EXIT");
   printf("\n\n\t\tEnter Your Option :--> ");
   scanf("%d",&option);
   switch(option)
   {
      case 1: add();
                 break;
      case 2: display();
                break;
      case 3: search();
                 break;
      case 4: modify();
                 break;
      case 5: delete();
                 break;
      default: printf("\n\t\tYou Pressed wrong key");
                 printf("\n\t\tProgram terminated");
                 exit(0);
   }
}
return 0;
}



// =========ADDING NEW RECORD========================

void add()
{
  FILE *fp;
   if((fp=fopen("studentInfo.txt","ab"))==NULL)     //binary file
   {
           //Print error message and return to main program
      }
   //  Accept the new student details from the user as elements of student structure
       ...........
       .............
    fwrite(&s,sizeof(s),1,fp);     // write the  structure variable to the file

    //check if the user wants to add another record
     ...................
  close the file
}


//==================DELETING A RECORD FROM LIST ============
void delete()
{
 FILE *fp;
if((fp=fopen("studentInfo.txt","rb"))==NULL)
{
       // Print error message and return to main program
 }

//Create a student tructure variable
.....................
....................
FILE *ft;

if((ft=fopen("temp.txt","wb"))==NULL)    // temporary file
{
   // Print error message and return to main program
}
// Read the roll no. of the student
//Loop through the student file reading one record into structure variable till end of file is
reached
{
       ........................
           // check if roll no is not matching
                    fwrite(&s,siz,1,ft);  // write the record to temporary file
         //  if match found,  skip writing this record and
                    // continue reading next record
}
//close the files
remove("studentInfo.txt"); //remove old student file
rename("temp.txt","studentInfo.txt");  //rename temporary file as student file
```

```
//if record not found
    // print " ERROR RECORD NOT FOUND "
}

//===========MODIFY A RECORD ===========================
void modify()
{
FILE *fp;
if((fp=fopen("studentInfo.txt","rb"))==NULL)
{
        // Print error message and return to main program
}
// Create a structure variable
// read the roll no. of the student
...........
................
................
while((fread(&s,siz,1,fp))==1)  //Loop through the file until end of file is reached
{

         //check if the roll no. exists in the file
            .................
        }
}

if(flag==1)  //  if found
    {
       .........................
        //accept the new details of the student as structure elements
      .......................
     // position the cursor to the current record position using fseek() function
    fwrite(&s,sizeof(s),1,fp); // write the structure to the file
}

else printf("\n\n\t!!!! ERROR !!!! RECORD NOT FOUND");
//close the file
}
//===================DISPLAY THE LIST =================
void display()
{
     FILE *fp;
    if((fp=fopen("studentInfo.txt","rb"))==NULL)
```

```
     {
             // Print error message and return to main program
      }
// create a student structure variable
.......................
.......................
// Loop through the student file reading one record at a time into structure variable till end
of file //is reached

while((fread(&s,siz,1,fp))==1)
    {
          //print the student details on the screen
              ...............
               ................
    }
   close the file
}

void search()
{
    FILE *fp;
    if((fp=fopen("studentInfo.txt","rb"))==NULL)
     {
              printf("can't open file");
              return;
      }
          printf("\n\n\tEnter Roll Number of Student to search the record : ");
          scanf("%d",&tempRoll);

          //Loop through the student file reading one record into structure variable until EOF is
              reached
         while(. .................)
         {
                    //compare roll no.'s
                   // if match found
                          // display student details
                      .......................
                       ..................
             }
else print "!!!! ERROR RECORD NOT FOUND !!!!"
//close the file
..........

}
```

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES

Case Study 2:
Library Management System

"Library Management System" is used by education establishments to manage books in library. A librarian can add, search, edit and delete books. This section is password protected. That means you need administrative credentials to log in as a librarian. The credentials are created when the program is executed from the first time. A student can search for the book and check the status of the book if it is available.

Algorithm:

```
Step1: Create a structure for the Library Login
Step2:  Print  "Log in as"
           Print "1. Librarian"
           Print "2. Student "
           Print "3. Exit"
           Print "Enter your choice: "
Step3:  Accept the user's choice.
Step4:  If choice = 1, enter login and password of Librarian
Step 5:After Login in successful
           Print  "Login Successful"
       Print  "Add Book"
       Print "2: Search Book"
       Print "3: Edit Book"
       Print "4: Delete Book"
       Print "5: Moderate Student Request"
       Print "Enter your choice: "
Step 6: If choice = 1 Provide the following information for title,Author,ISBN, category,
           Publication ,description , Increment Id by 1 and save the record
           If choice = 2 Enter the book id to search ,
                  if Id match book information has to display
                  else
```

```
        print book id did'nt match
    If choice = 3 Enter the book id to edit
        If book ID match and has edit all fields of book
        Else
        Print Sorry!! The book is not in the database
    If choice = 4 Enter the book id to Delete
        If book ID match and has to delete fields of book
        Else
        Print  Sorry!! The book is not in the database
```

Step 7: if choice = 2 Enter the book id to search

```
        if Id match book information will display
            print "Status: "
            if  book taken
            print "Not Available"
             else
             print "Available"
```

Step 8: if choice = 3 then exit

Sample Code:

```
#include < stdio.h >
#include < stdlib.h >
#include < string.h >
void flush() {
    int c;
    while ((c = getchar()) != '\n' & amp; & amp; c != EOF);
  }
typedef struct {
  char username[15];
  char password[15];
}User;

typedef struct {
  int id; // must be unique
  char title[50];
  char author[50]; // if more than two, separate using ,(COMMA)
  char ISBN[50];
  char category[50];
  char publication[50];
  char description[255];
  int taken;
}
Book;

int main() {
  char option, admin_option, username[15], password[15], edit_option;
  int first_time, c, i, j, id, found;
  FILE * f;
  User user;
  Book book;

  f = fopen("librarian.check", "r");
  if (f == NULL) {
    fclose(f);
    f = fopen("librarian.check", "w");
    fputc(1, f);
    fclose(f);
  } else {
    fclose(f);
  }
```

```
    f = fopen("id.check", "r");
    if (f == NULL) {
      fclose(f);
      f = fopen("id.check", "w");
      fputc(0, f);
      fclose(f);
    } else {
      fclose(f);
    }
  //Display the menu of Logins of Librarian ,student and exit
     scanf("%c", & amp; option);
    switch (option) {
    case '1':
      f = fopen("librarian.check", "r");
      if (f == NULL) {
        printf("Couldn't read file\n");
        exit(0);
      } else {
        first_time = fgetc(f);
        if (first_time == 1) {
        fclose(f);
          flush();
          printf("Provide username and password to setup\n");
           // Read the username and password
          // write this credential to file
          if (f == NULL) {
            //Print error message and exit
          }
          fwrite(user, sizeof(User), 1, f);
          fclose(f);
          printf("Exit and login again to continue\n");
          f = fopen("librarian.check", "w");
          fputc(0, f);
          fclose(f);
        } else {
          fclose(f);
          flush();
          printf("Provide credential to login\n");
          // Read the username and password

          // read the credential from file

          if (f == NULL) {
               //Print error message and exit
          }
          fread(user, sizeof(User), 1, f);
          if (strcmp(username, user.username) != 0 || strcmp(password, user.password) != 0) {

              //Print invalid username and password ,exit

          }
          printf("Login Successful!!\n");
          printf("1: Add Book\n");
          printf("2: Search Book\n");
          printf("3: Edit Book\n");
          printf("4: Delete Book\n");
          printf("4: Moderate Student Request\n");
          printf("Enter your choice: ");
          scanf("%c", & amp; admin_option);
          switch (admin_option) {
          case '1':
            flush();
            printf("Provide the following information\n");

     //read the inforamtion of TITLE,AUTHOR,ISBN,CATEGORY,PUBLICATION AND DESCRIPTION

            // increment ID
            f = fopen("id.check", "r");
            if (f == NULL) {

                      //Print error message and exit

            }
            id = fgetc(f);
            fclose(f);
            book.id = id;

            // save the record

          case '2':
            printf("Enter the book id to search: ");
            scanf("%d", & amp; id);

            // search in the database
            f = fopen("book.record", "rb");
            found = 0;
            i = 0;
            while (fread(book, sizeof(Book), 1, f)) {
```

```
      if (book.id == id) {

        // Display matched  data books


      }
      i++;
    }
    if (found == 0) {

                          //Print error message and exit

    }
    fclose(f);
  break;
  case '3':
    printf("Enter the book id to edit: ");

                //Read the book id


    f = fopen("book.record", "rb+");
    found = 0;
    while (fread( & amp; book, sizeof(Book), 1, f)) {
      if (book.id == id) {
        // matched
        found = 1;
        break;
      }
    }
    if (found == 0) {


                //Print error message

    } else {
      printf("What field do you want to edit:\n");
      printf("1. Title\n");
      printf("2. Author\n");
      printf("3.  ISBN\n");
      printf("4. Category\n");
      printf("5. Publication\n");
      printf("6. Description\n");
      printf("Enter your choice: ");
```

```
scanf("\n%c", & amp; edit_option);
switch (edit_option) {
case '1':
  flush();
  printf("Enter new title: ");

        //Read the book title

  break;
case '2':
  printf("Enter new author: ");

        //Read the author

  break;
case '3':
  printf("Enter new ISBN: ");


        //Read the ISBN number

  break;
case '4':
  printf("Enter new Category: ");

        //Read the category

  break;
case '5':
  printf("Enter new Publication: ");

        //Read the publication

  break;
  printf("Enter new Description: ");

        //Read the Description

  break;
default:
  printf("Enter 1 to 6\n");
  break;
}
fseek(f, i, SEEK_SET);
```

```
      fwrite( & amp; book, sizeof(Book), 1, f);
      fclose(f);

                  //print modified database message
    }
    break;
  case '4':
    printf("Enter the book id to Delete: ");

      //read the id and open the record



    found = 0;
    while (fread( & amp; book, sizeof(Book), 1, f)) {
      if (book.id == id) {
        // matched
        found = 1;
        break;
      }
    }
    fclose(f);
    if (found == 0) {
      printf("Sorry!! The book is not in the database\n");
    } else {
      // create a temporary file
      FILE * temp;
      temp = fopen("book.temp", "a");
      if (temp == NULL) {

                  //print the error message

      }

      f = fopen("book.record", "rb");


      // copy all the items except item to delete
      // to temporary file


      }
      fclose(f);
      fclose(temp);
```

```
          // delete original file


          // rename the temporary file

          printf("Book record deleted from the database!!\n");
         }
         break;
        default:
         printf("Enter only 1 - 4\n");
         break;
        }
       }
      }
      break;
     case '2':
      printf("Enter the book id to search: ");
      scanf("%d", & amp; id);

      // search in the database


      while (fread( & amp; book, sizeof(Book), 1, f)) {
       if (book.id == id) {


                    // matched display the book information


        printf("Status: ");
        if (book.taken == 1) {
         printf("Not Available");
        } else {
         printf("Available\n");
        }
        break;
       }
       i++;
      }
      if (found == 0) {
       printf("Sorry!! The book is not in the database\n");
      }
      fclose(f);
      break;
     case '3':
      printf("Bye!!\n");
      exit(0);
     default:
      printf("Enter either 1 or 2 only\n");
      break;
     }
   }
```

RECORD NOTES

RECORD NOTES

RECORD NOTES

RECORD NOTES