Lab Manual

for

Programming in C Lab

2139

Diploma In Computer Engineering

2nd Semester

By

SITTTR
Kalamassery

STATE INSTITUTE OF TECHNICAL TEACHERS TRAINING AND RESEARCH

GENERAL INSTRUCTIONS

Rough record and Fair record are needed to record the experiments conducted in the laboratory. Rough records are needed to be certified immediately on completion of the experiment. Fair records are due at the beginning of the next lab period. Fair records must be submitted as neat, legible, and complete.

INSTRUCTIONS TO STUDENTS FOR WRITING THE FAIR RECORD

In the fair record, the index page should be filled properly by writing the corresponding experiment number, experiment name , date on which it was done and the page number.

On the **right side** page of the record following has to be written:

1. **Title**: The title of the experiment should be written in the page in capital letters.
2. In the left top margin, experiment number and date should be written.
3. **Aim**: The purpose of the experiment should be written clearly.
4. **Apparatus/Tools/Equipments/Components used:** A list of the Apparatus/Tools /Equipments /Components used for doing the experiment should be entered.
5. **Principle**: Simple working of the circuit/experimental set up/algorithm should be written.
6. **Procedure:** steps for doing the experiment and recording the readings should be briefly described(flow chart/programs in the case of computer/processor related experiments)
7. **Results**: The results of the experiment must be summarized in writing and should be fulfilling the aim.
8. **Inference** : Inference from the results is to be mentioned.

On the **Left side** page of the record following has to be recorded:

1. **Circuit/Program**: Neatly drawn circuit diagrams/experimental set up.
2. **Design**: The design of the circuit/experimental set up for selecting the components should be clearly shown if necessary.
3. **Observations:** i) Data should be clearly recorded using Tabular Columns.

ii) Unit of the observed data should be clearly mentioned

iii) Relevant calculations should be shown. If repetitive calculations are needed, only show a sample calculation and summarize the others in a table.

4. **Graphs :** Graphs can used to present data in a form that show the results obtained, as one or more of the parameters are varied. A graph has the advantage of presenting

large

amounts of data in a concise visual form. Graph should be in a square format.

GENERAL RULES FOR PERSONAL SAFETY

1. Always wear tight shirt/lab coat , pants and shoes inside workshops.

2. REMOVE ALL METAL JEWELLERY since rings, wrist watches or bands, necklaces, etc. make excellent electrodes in the event of accidental contact with electric power sources.

3. DO NOT MAKE CIRCUIT CHANGES without turning off the power.

4. Make sure that equipment working on electrical power are grounded properly.

5. Avoid standing on metal surfaces or wet concrete. Keep your shoes dry.

6. Never handle electrical equipment with wet skin.

7. Hot soldering irons should be rested in its holder. Never leave a hot iron unattended.

8. Avoid use of loose clothing and hair near machines and avoid running around inside lab .

TO PROTECT EQUIPMENT AND MINIMIZE MAINTENANCE:

DO: 1. SET MULTIRANGE METERS to highest range before connecting to an unknown source.

2. INFORM YOUR INSTRUCTOR about faulty equipment so that it can be sent for repair.

DO NOT: 1. Do not MOVE EQUIPMENT around the room except under the supervision of an instructor.

Table of Contents

C LANGUAGE

INTRODUCTION

C is a programming language developed at AT&T's BELL Laboratory of USA in 1972. Dennis Ritchie designed it. Because of its reliability. C is very popular. C is highly portable & it is well suited for structured programming. C program consists of collection of functions.

Hardware Requirement : Desktop Computer / labtop computer

Software Requirement : Linux Operating System with GCC/ TURBOC IN WINDOWS OS

GCC

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, so if you are running Unix or a Linux variant you likely have it on your system. You can invoke gcc on a source code file simply by typing :-

gcc *filename*

The default executable output of gcc is "a.out", which can be run by typing " ./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*

Again, you can run your program with "./outputfile". (The ./ is there to ensure you run the program for the current working directory.)

Note: If you need to use functions from the math library (generally functions from math.h such as sin or sqrt), then you need to explicitly ask it to link with that library with the -l flag and the library 'm':

gcc *filename* -o **outputfile** –lm

Turbo C/C++

Open Turbo C from your Desktop or Programs menu. Select **"File"** from Menu bar and select option **"New" and** Save C program in filename .C extension.

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.

To RUN the program – you may **select ->Run** from menu and **click -> Run**

Now you will see the output screen.

Experiment 1

Evaluate area of triangle with 3 sides given.

Aim : To evaluate area of a triangle using the formula area=sqrt(s(s-a)(s-b)(s-c)) where a,b,c are the sides of the triangle and s=(a+b+c)/2

Objectives : Study about basic building blocks such as **c**onstants, variables, keywords, operators, expressions and input output statements in C language.

Tools/Equipents : Editors like gedit,vi in linux with gcc (or TurboC editor in windows), esktop or laptop computer.

Algorithm

Step1:start

Step2:input a,b,c

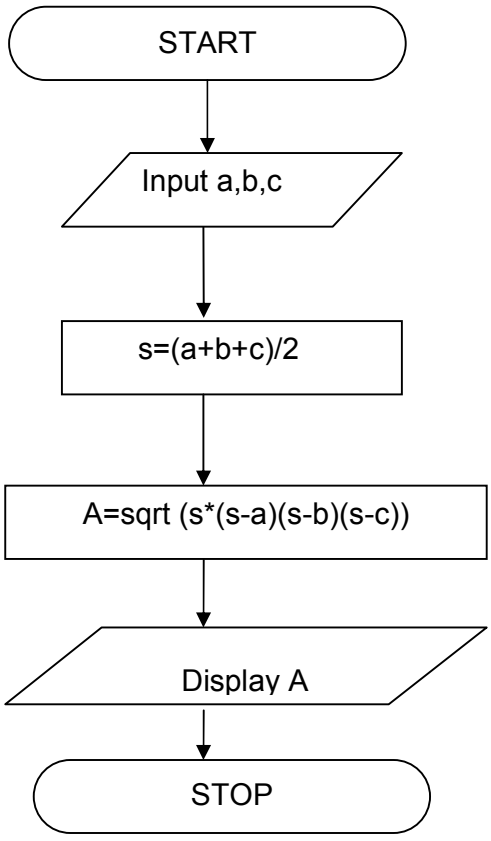
Step3:s=(a+b+c)/2

Step4:A=sqrt(s*(s-a)(s-b)(s-c))

Step5:PrintA

Step 5:stop

Flowchart



Procedure

Save program in a file, Compile program, debug errors, Execute or Run program with necessary inputs. Verify the outputs obtained.

Program

```
/*To evaluate area of triangle (sqrt(s(s-a)(s-b)(s-c)*/
#include<math.h>
#include<stdio.h>
void main()
{
int a,b,c;
float s,area;
clrscr();
printf("enter the values of a,b,c");
scanf("%d%d%d",&a,&b,&c);
s=(a+b+c)/2.0;
area=sqrt(s*(s-a)*(s-b)*(s-c));
printf("the area of a trangle is =%f",area);
getch();
}
```

Input Output

```
enter the values of a,b,c
15
20
30
The area of a trangle is = 133.317
```

Result The program is compiled, executed and the output is verified

Sample Questions

1. Write a C program to find the area of a circle using the formula: Area = PI * r^2

2. Write a C program to find the area and volume of sphere. Formulas are:-
   Area = 4*PI*R*R Volume = 4/3*PI*R*R*R.
   Note : Assume PI=3.14

3. Write a C program to print the multiply value of two accepted numbers

4. Write a C program to convert centigrade into Fahrenheit. Formula: C= (F-32)/1.8.

5. Write a C program to read in a three digit number and produce the following output (assuming that the input is 347)

3 hundreds
4 tens
7 units

6. Write a C program to read in two integers and display one as a percentage of the other. Typically your output should look like

20 is 50.00% of 40
assuming that the input numbers were 20 and 40. Display the percentage correct to 2 decimal places.

7. Write a C program to find out whether the character pressed through the keyboard is a digit or not (using conditional operator).

8. Write a C program to swap variable values of i and j.

9. Write a program to read a floating point number from keyboard and print its integer and fractional part separately.

10. Write a program to read and print a character using getchar(), getch(), getche(), putchar(), putch().

11. Write a program to find the Simple Interest(Simple Interest,I=P*N*R/100) where P=Principal,N=no: of years, R= Rate of Interest.

Experiment 2

Write a program to find greatest among 3 numbers

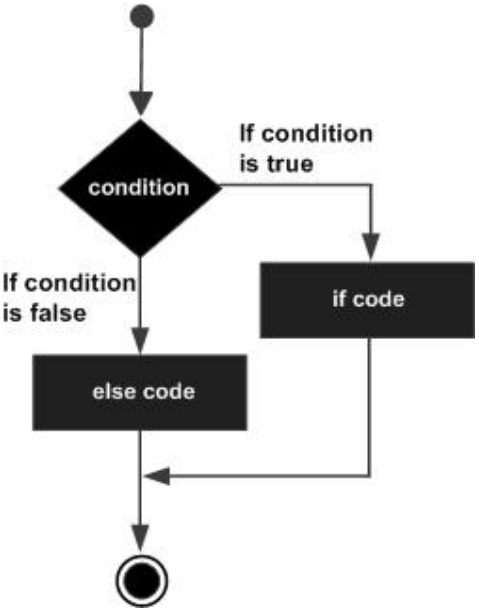
Aim : Program to find greatest among 3 numbers

Objectives : Study about decision statements such as' if else' statement that allow us to choose to execute one sequence of instructions over one or more others depending on certain circumstances.

Tools/Equipents : Editors like gedit,vi editor in linux with gcc (or TurboC editor in windows) in desktop or laptop computer.

Theory : If the Boolean expression evaluates to **true**, then the **if block** will be executed, otherwise, the **else block** will be executed.

C programming language assumes any **non-zero** and **non-null** values as **true**, and if it is either **zero** or **null**, then it is assumed as **false** value.



The **if else ladder** statement in C programming language is used to test set of conditions in sequence. An if condition is tested only when all previous if conditions in if-else ladder is false. If any of the conditional expression evaluates to true, then it will execute the corresponding code block and exits whole if-else ladder.

Syntax

```
if(condition_expression_One) {
   statement1;
} else if (condition_expression_Two) {
   statement2;
} else if (condition_expression_Three) {
```

```
   statement3;
} else {
   statement4;
}
```

- *First of all condition_expression_One is tested and if it is true then statement1 will be executed and control comes out out of whole if else ladder.*
- *If condition_expression_One is false then only condition_expression_Two is tested. Control will keep on flowing downward, If none of the conditional expression is true.*
- *The last else is the default block of code which will gets executed if none of the conditional expression is true.*

Algorithm

Step1:start

Step2:input a,b,c

Step3:if(a>b) and (a>c) go to step 4 otherwise go to step 5

Step4: display a is greater go to 8

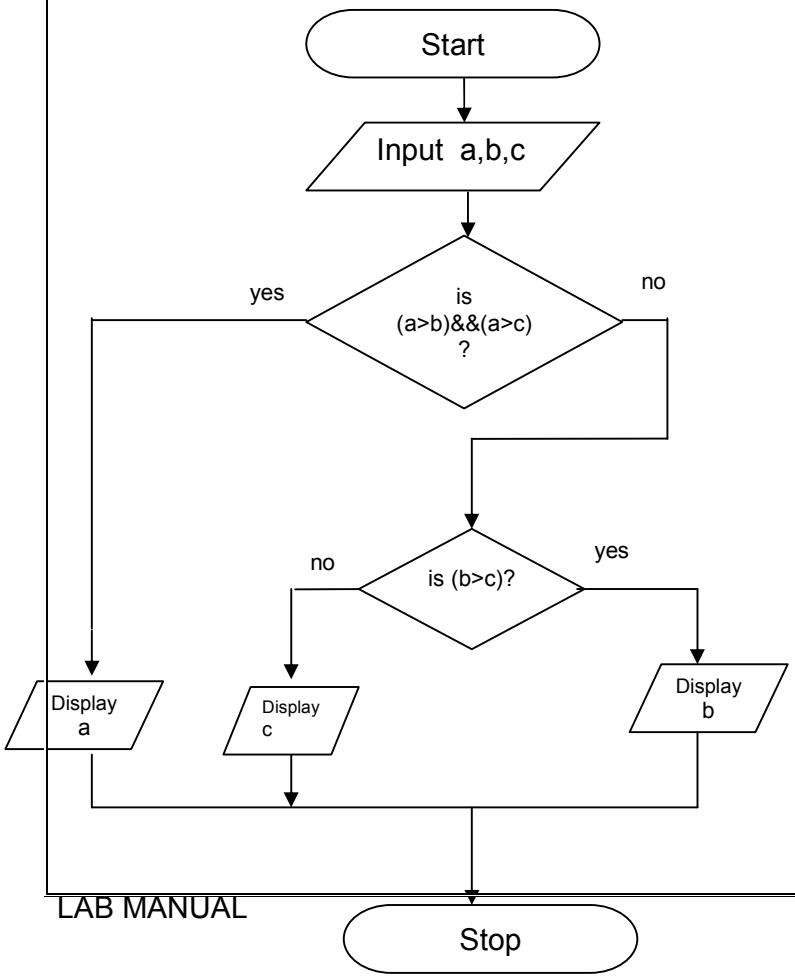
Step5: if (b>c) go to step 6 otherwise step 7

Step 6: display b is greater, go to 8

Step 7: display c is greater

Step 8:stop

Flowchart



Procedure

Save program in a file, Compile program, debug errors, Execute or Run program with necessary inputs. Verify the outputs obtained.

Program

```
/*Program to find greatest among 3 numbers*/
void main()
{
int a,b,c;
clrscr();
printf("enter the values of a,b and c");
scanf("%d%d%d",&a,&b,&c);
if(a>b && a>c)
printf("%d is greatest of  %d  %d  %d", a,a,b,c);
else
if(b>c)
printf("%d is greatest of  %d  %d  %d",b,a,b,c);
else
printf("%d is gratest of %d  %d  %d",c,a,b,c);
getch();
}
```

Input Output

Enter the values of a,b and c

10

30

20

30 is greatest of 10 30 20

Result: The program is compiled , executed and the output is verified

Sample Questions

1. Write a C program to find that the accepted number is Negative, Positive or Zero.

2. Write a program which reads two integer values. If the first is lesser print the message up. If the second is lesser, print the message down if they are equal, print the message equal; if there is an error in reading the data, print a message containing the word Error.

3. Write a C program that prints the given three integers in ascending order using if – else.

4. Given three integers as input representing a date as day, month, year, print the number day, month and year for the next day's date.

Typical input: “28 2 1992” Typical output: “Date following 28:02:1992 is 29:02:1992”.

5. Write a program to find the roots of a quadratic equation.

6.In a shop, all the items are grouped into 4 different categories, each group has different percentage of discount 5%,10%,15%,20% . Compute price, category, discount and net price.(Use Menu generation)

7. Write a program to grade the student according to following rule.

Marks	Grade
70 to 100	Distinction
60 to 69	First class
50 to 59	Second class
40 to 49	Pass class
0 to 39	Fail

Experiment 3

Write a Program to perform the arithmetic expression using switch statement

Aim: Program to perform the arithmetic expression using switch statement

Objectives: Study about -decision control statement such as' switch' statement.

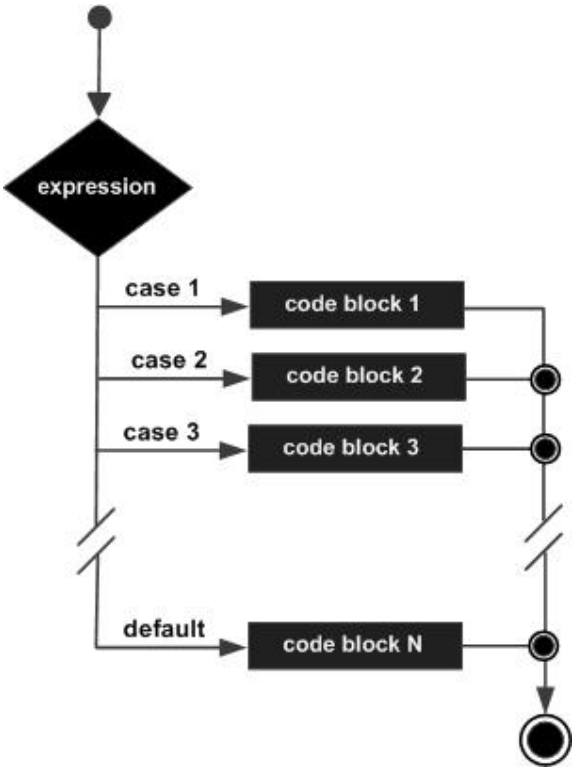
Tools/Equipents : Editors like gedit,vi editor in linux with gcc (or TurboC editor in windows) in desktop or laptop computer.

Theory: A **switch** statement allows a variable(/expression) to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each **switch case**. It's similar to if else if ladder statement.

The **expression** used in a **switch** statement must have an integral or enumerated type, or be of a class type in which the class has a single conversion function to an integral or enumerated type.

You can have any number of case statements within a switch. Each case is followed by the value to be compared to and a colon. The default block is written normally at the end of switch structure

The **constant-expression** for a case must be the same data type as the variable in the switch, and it must be a constant or a literal.



Note : A break statement must included at the end of each case block to perform the execution as per the logic shown in the flow chart above

Algorithm

Step1: start

Step2: input a,b,op

Step3: switch(op)

Step4: case '+': print 'sum of a& b is' ,a+b

Step5: case '-': print 'difference of a& b is' ,a-b

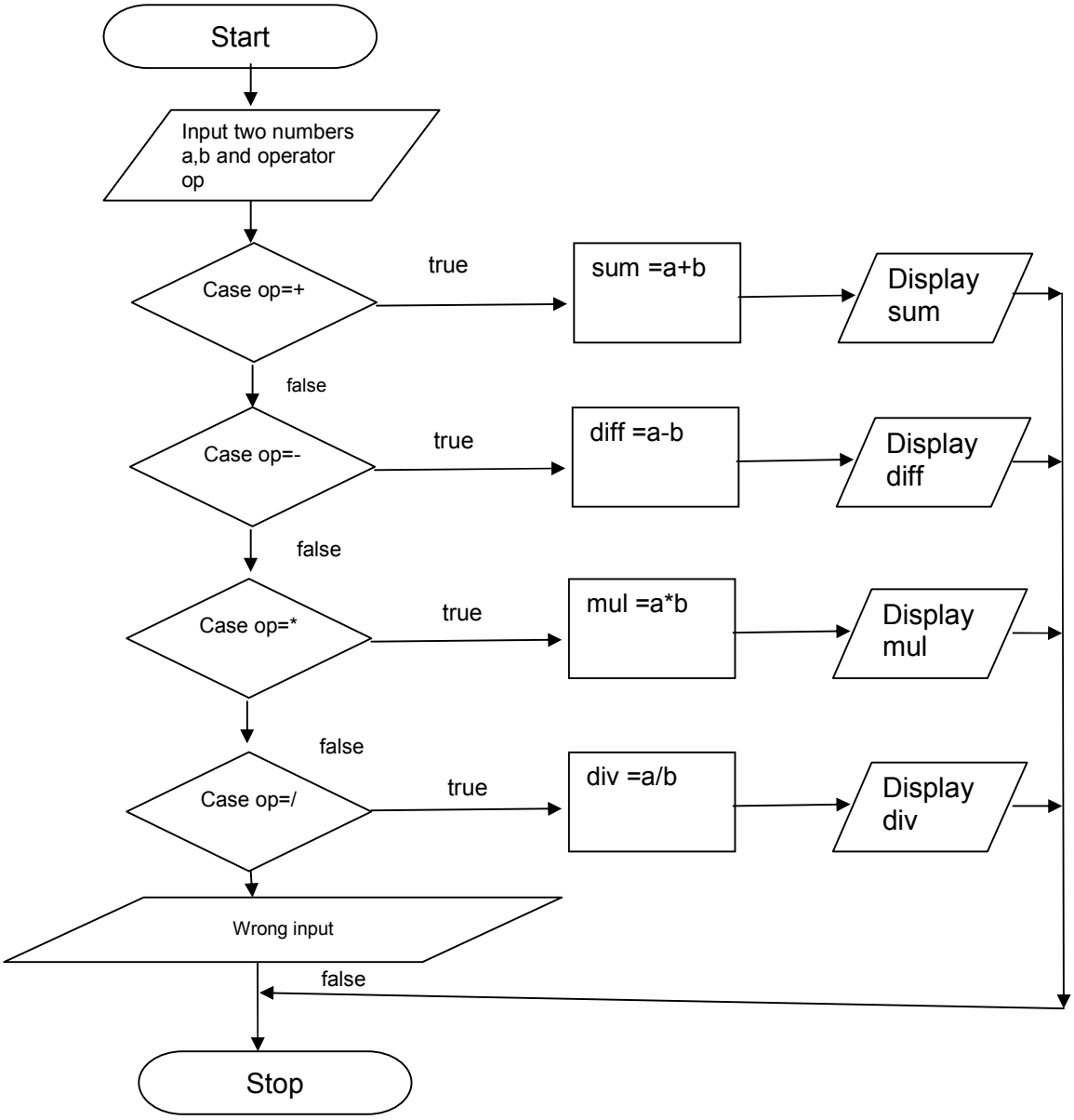
Step6: case '*': print 'product of a& b is', a*b

Step7: case '/': print 'quotient of a& b is', a/b

Step8: default: invalid option

Step9: stop

Flowchart:



Procedure

Save program in a file, Compile program, debug errors, Execute or Run program with necessary inputs. Verify the outputs obtained.

Program

/*Program to perform the arithmetic expression using switch statement*/

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,b;
int op;
clrscr();
printf(" 1.addition\n 2.subtraction\n 3.multiplication\n 4.division\n");
printf("enter the values of a & b");
scanf("%d%d",&a,&b);
printf("enter your choice : ");
scanf("%d",&op);
switch(op)
{
case 1      :printf("sum of %d and %d=%d",a,b,a+b);
break;
case 2      :printf("difference of %d and %d=%d",a,b,a-b);
break;
case 3      :printf("multiplication of %d and %d=%d",a,b,a*b);
break;
case 4      :printf("Division of two numbers is %d=",a/b);
break;
default     : printf(" Enter Your Correct Choice.");
}
getch();
}
```

Input Output

1. Addition

2. Substraction

3. Multiplication

4. Division

Enter your choice : 1

Enter a and b values 10 20

Sum of 10 and 20 = 30

Result: The program is compiled, executed and the output is verified.

Sample Questions:

1.Display weekday in word by giving weekday in integer
2.Calculate total days in a month and year
3.Display month in word format on giving month in integer format.

Experiment 4

Write a program to find the factorial of a given number

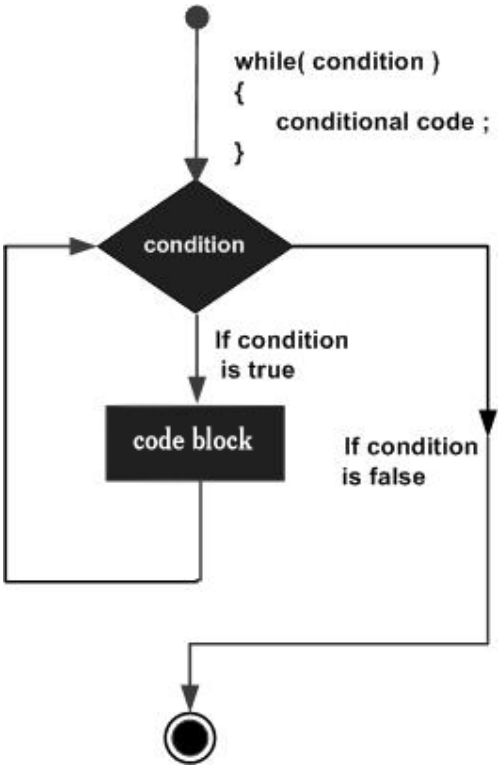
Aim: Program to find the factorial of a given number

Objectives: Study about iteration control statement such as' while' statement.

Tools/Equipents: Editors like gedit,vi editor in linux with gcc (or TurboC editor in windows) in desktop or laptop computer.

Theory: A **while** loop in C programming repeatedly executes a set of target statements as long as a given condition is true.

When the condition becomes false, the program control passes to the line immediately following the loop.



Algorithm

Step1: start

Step2: input n,i,f

Step3: Initialize f=i=1

Step4: if(i<=n) repeat steps 5 and 6 otherwise go to step 7

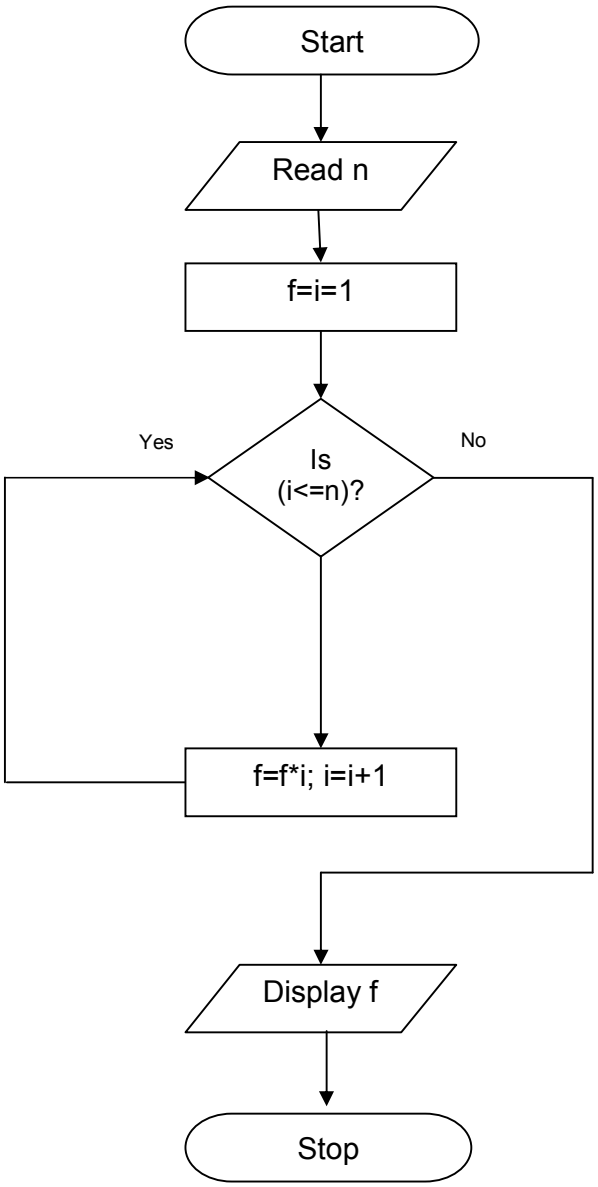
Step5: f=f*i

Step6: i=i+1

Step7: print f

Tep8: stop

Flowchart



Procedure

Save program in a file, Compile program, debug errors, Execute or Run program with necessary inputs. Verify the outputs obtained.

Program

```
/*Program to  find the factorial of a given number*/
void main()
{
int n,i,f;
f=i=1;
clrscr();
printf("enter a number");
scanf("%d",&n);
while(i<=n)
{
f*=i;
i++;
}
printf("the factorial of %d is %d",n,f);
getch();
}
```

Input Output

Enter a number 5

The factorial of 5 is 120

Result: The program is compiled, executed and the output is verified.

Sample Questions

Experiment 5

Write a program to generate all prime numbers up to nth number.

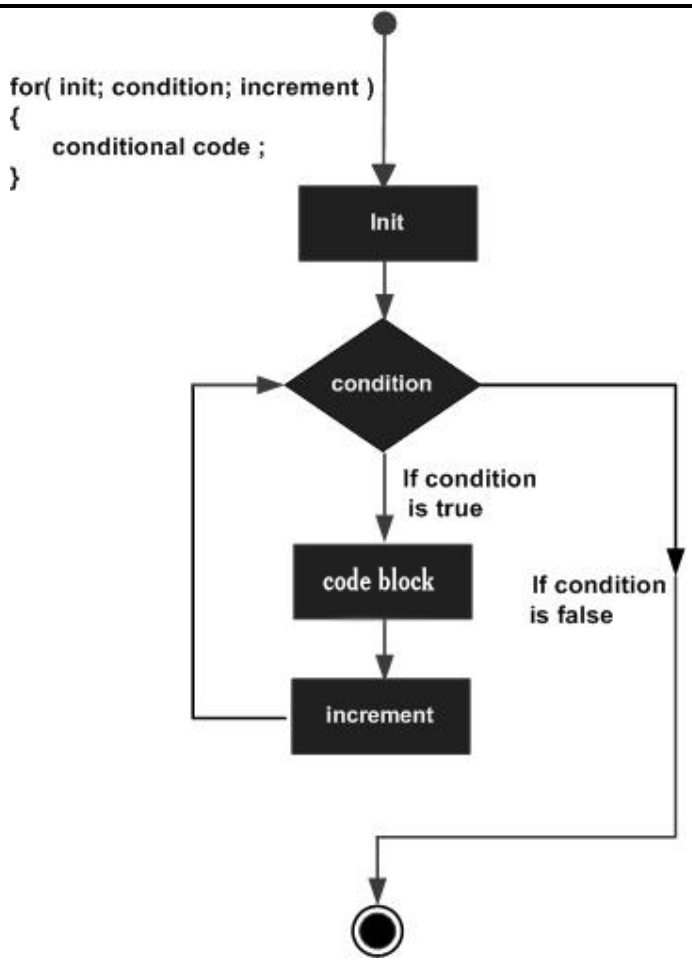
Aim: Program to prime numbers till nth number

Objectives: Study about iteration control statement such as' for' statement.

Tools/Equipents: Editors like gedit,vi editor in linux with gcc (or TurboC editor in windows) in desktop or laptop computer.

THEORY: A **for** loop is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times.

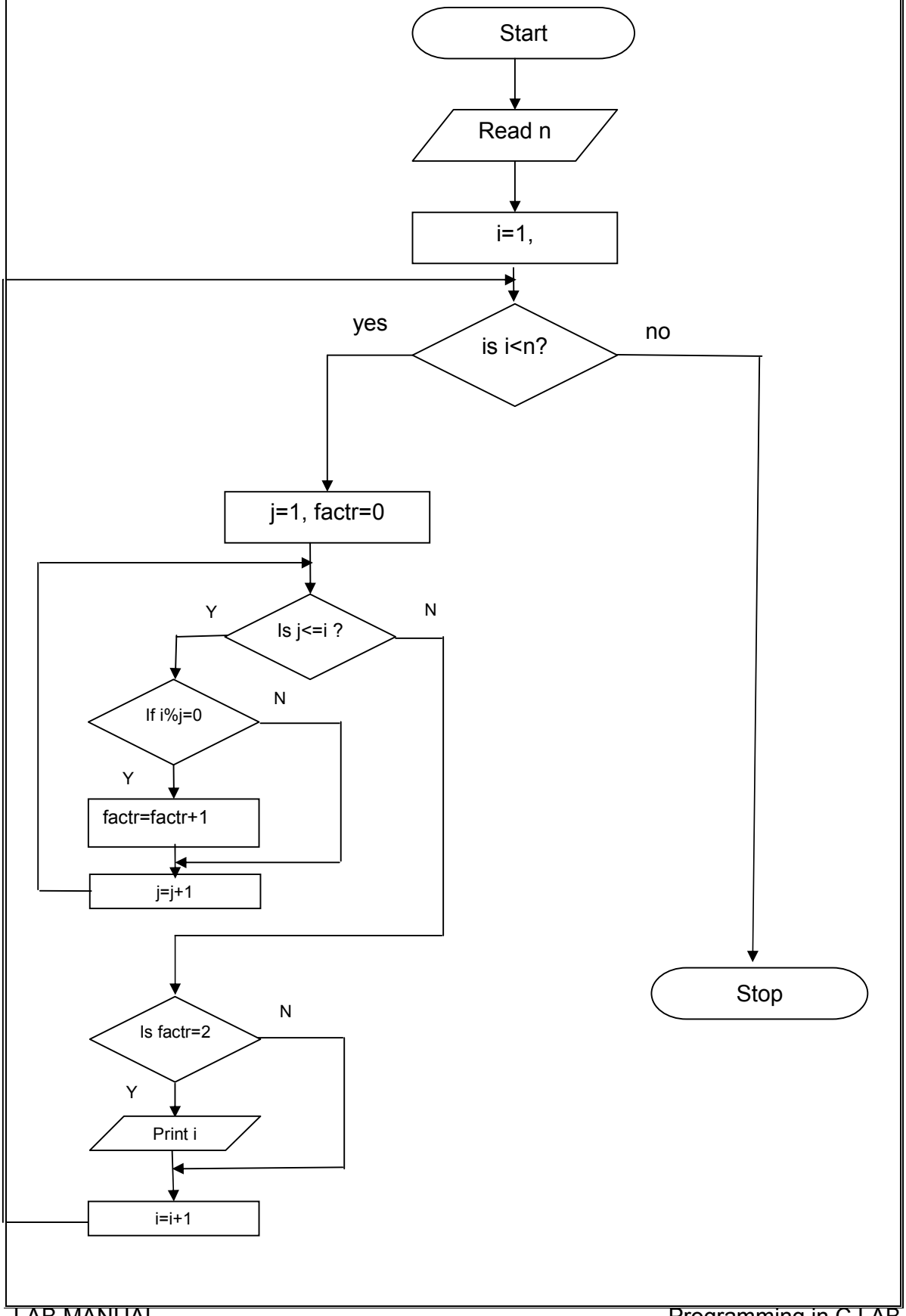
- The **init** step is executed first, and only once. This step allows you to declare and initialize any loop control variables. You are not required to put a statement here, as long as a semicolon appears.
- Next, the **condition** is evaluated. If it is true, the body of the loop is executed. If it is false, the body of the loop does not execute and the flow of control jumps to the next statement just after the 'for' loop.
- After the body of the 'for' loop executes, the flow of control jumps back up to the **update** statement. This statement allows you to update any loop control variables. This statement can be left blank, as long as a semicolon appears after the condition.
- The condition is now evaluated again. If it is true, the loop executes and the process repeat**s** itself (body of loop, then update step, and then again condition). After the condition becomes false, the 'for' loop terminates.



Algorithm

Step 1: start
Step 2: read n
Step 3: set i=1
Step 4: if (i<=n) go to step 5 otherwise got to step 13
Step 5: set factr = 0
Step 6: set j = 1
Step 7: if(j<=i) go to step 8 otherwise go to step 11
Step 8: if(i mod j ==0) go to step 9 otherwise go to step 10
Step 9 factr =factr+1
Step 10: j=j+1,go to step 7
Step 11: if factr = 2, print i as prime number
Step 12: i=i+1 ,goto step 4
Step 13: stop

Flowchart


Start
Read n
i=1,
yes
is i<n?
no
j=1, factr=0
Y
Is j<=i ?
N
If i%j=0
N
Y
factr=factr+1
j=j+1
Stop
Is factr=2
N
Y
Print i
i=i+1

Procedure

Save program in a file, Compile program, debug errors, Execute or Run program with necessary inputs. Verify the outputs obtained.

Program

```
/*Program to generate prime numbers till nth number*/
void main()
{
int n,i,factr,j;
printf("enter the range");
scanf("%d",&n);
printf("Prime numbers are\n");
for(i=1;i<=n;i++)
{
factr=0;
for(j=1;j<=i;j++)
{
if(i%j==0)
factr++;
}
if(factr==2)
printf("%d  ",i);
}
getch();
}
```

Input Output

```
Enter the range 10
Prime numbers are
3  5  7
```

Result: The program is compiled, executed and the output is verified

Sample Questions

1) Write a C program to display first 25 Fibonacci nos.

2) Write a C program to find factorial of accepted nos.

3) Write a C program to find the sum of digits of accepted no.

Experiment 6

Program to print Fibonacci series

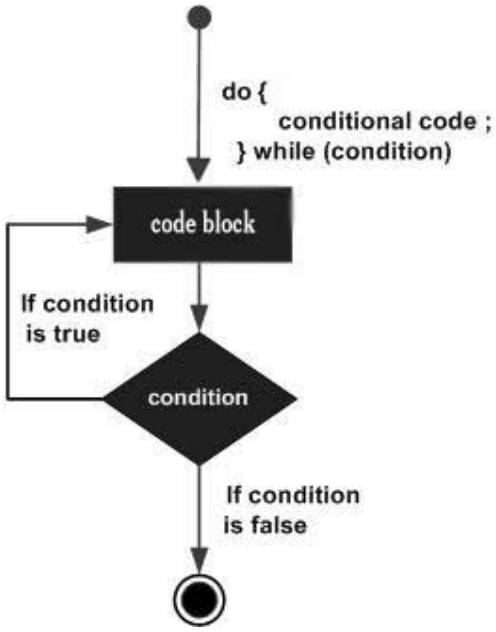
Aim: Program to print Fibonacci series

Objectives: Study about iteration control statement such as' do while' statement.

Tools/Equipents: Editors like gedit,vi editor in linux with gcc (or TurboC editor in windows) in desktop or laptop computer.

Theory: Unlike **for** and **while** loops, which test the loop condition at the top of the loop, the **do...while** loop in C programming checks its condition at the bottom of the loop.

A **do...while** loop is similar to a while loop, except the fact that it is guaranteed to execute at least one time.



Algorithm

Step 1: start

Step 2: set f=0,f1=0,f2=1,i=1

Step 3: read n

Step 4: Print f

Step 5: f=f1+f2

Step 6: Print f

Step 7:f2=f1

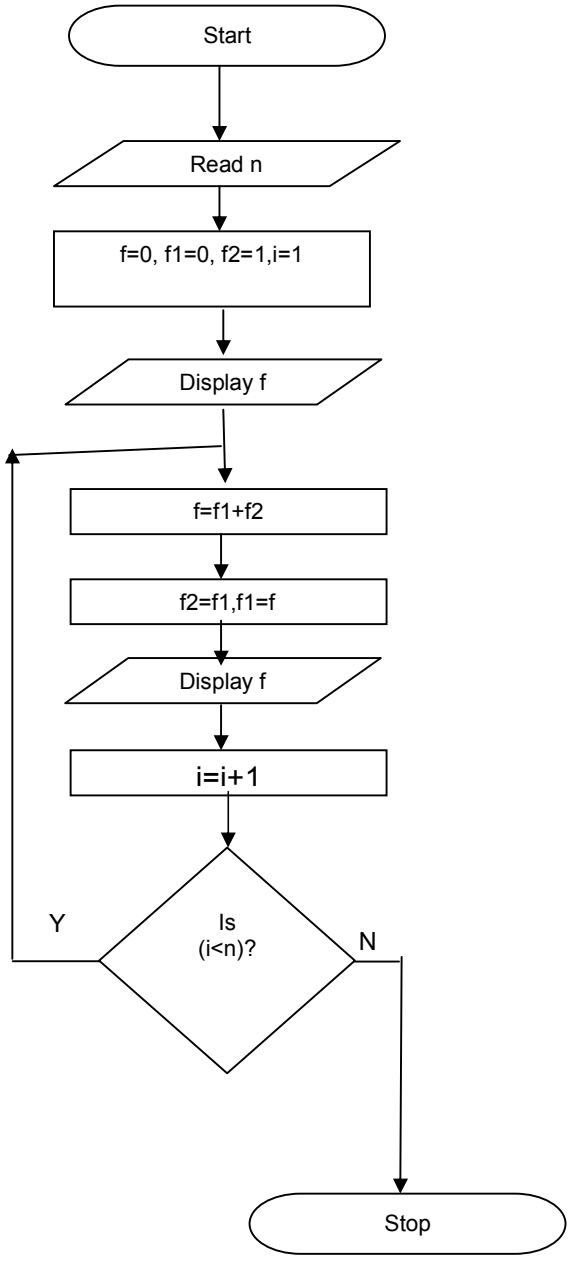
Step 8: f1=f

Step 9: i=i+1

Step 5: if (i<n) ,go to step 5 otherwise go to step10

Step 10: stop

Flowchart



Procedure

Save program in a file, Compile program, debug errors, Execute or Run program with necessary inputs. Verify the outputs obtained.

Program

```
/*Program to print Fibonacci series*/
void main()
{
int i=1,n,f,f1,f2;
printf("enter the range");
scanf("%d",&n);
f=0;
f1=0;
f2=1;
do
{
i++;
printf("%d\n",f);

f=f1+f2;
f2=f1;
f1=f;
}
while(i<=n);
}
```

Input Output

Enter the range 9

0 1 1 2 3 5 8 13 21

Result: The program is compiled, executed and the output is verified.

Sample Questions

1. Display the perfect numbers between 1 to N.
2. Write a menu driven program to perform arithmetic operation on given two numbers as per user choice.
3. Write a program to find the largest/smallest element in a given list of numbers.

Experiment 7

Program to find total of even integers

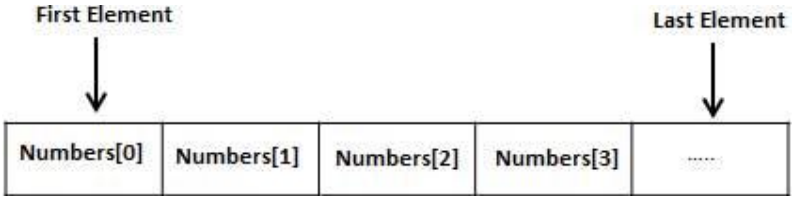
AIM: Program to find total of even integers in an array of n numbers.

Objectives: Study about arrays-one dimensional - in C language.

Tools/Equipents: Editors like gedit,vi editor in linux with gcc (or TurboC editor in windows) in desktop or laptop computer.

Theory: An array is a collection of variables of the same type.

All arrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest address to the last element.



To declare an array in C, a programmer specifies the type of the elements and the number of elements required by an array as follows :-

```
type arrayName [ arraySize ];
```

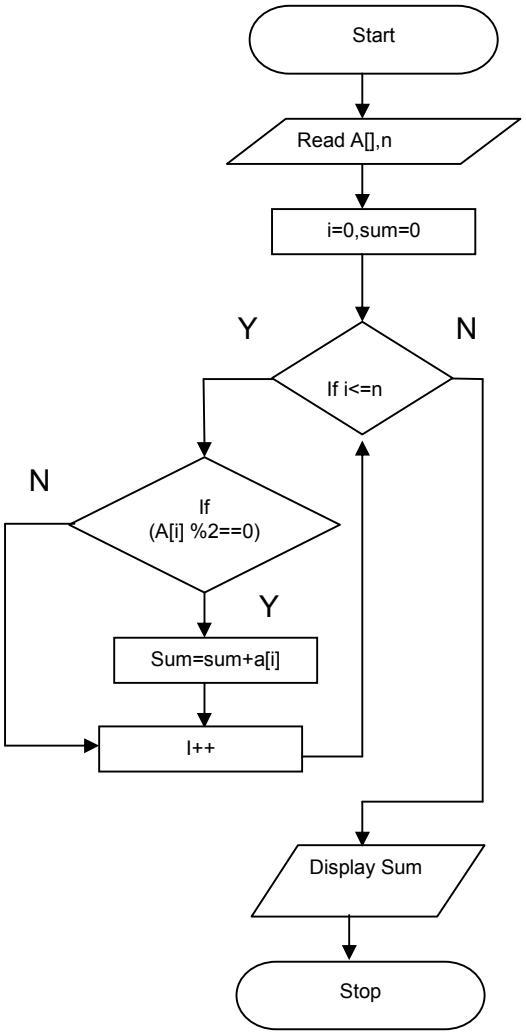
An element is accessed by indexing the array name. This is done by placing the index of the element within square brackets after the name of the array. For example :-

```
double salary = balance[9];
```

Algorithm

step1: start
Step 2: Read n and , n values to array A[]
Step3: initialize sum=0,i=0
Step 4: if(i<n) go to next step otherwise go to step 8
Step 5: if(A[i]%2==0) go to next step otherwise go to step 7
Step 6: sum=sum+A[i]
Step 7 i=i+1, go to step 4
Step8 print sum
Step 9 stop

Flowchart



Procedure:

Save program in a file, Compile program, debug errors, Execute or Run program with necessary inputs. Verify the outputs obtained.

Program

```
/*Program to  find total of even integers*/
#include<stdio.h>
main()
{
int a[20],i,sum=0;
printf("enter5 integers");
for(i=0;i<5;i++)
scanf("%d",&a[i]);
for(i=0;i<5;i++)
{
if(a[i]%2==0)
sum=sum+a[i];
}
prinf("sum =%d",sum);
getch();
}
```

Input Output

Entger 5 integers
2 4 7 8 2
Sum = 16

Result: The program is compiled, executed and the output is verified

Sample Questions

1.Program to implement bubble sorting for a set of integers.
2.Program to search an element in a list of numbers.

Experiment 8

Program to print product of two matrices

Aim: Program to print product of two matrices , if possible

Objectives: Study about arrays-two dimensional - in C language.

Tools/Equipents: Editors like gedit,vi editor in linux with gcc (or TurboC editor in windows) in desktop or laptop computer.

THEORY

The simplest form of multidimensional array is the two-dimensional array. A two-dimensional array is, a list of one-dimensional arrays. To declare a two-dimensional integer array of size [x][y], you would write something as follows :-

```
type arrayName [ x ][ y ];
```

Where **type** can be any valid C data type and **arrayName** will be a valid C identifier. A two-dimensional array can be considered as a table which will have x number of rows and y number of columns.

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Fig : A 3 x 4 Table

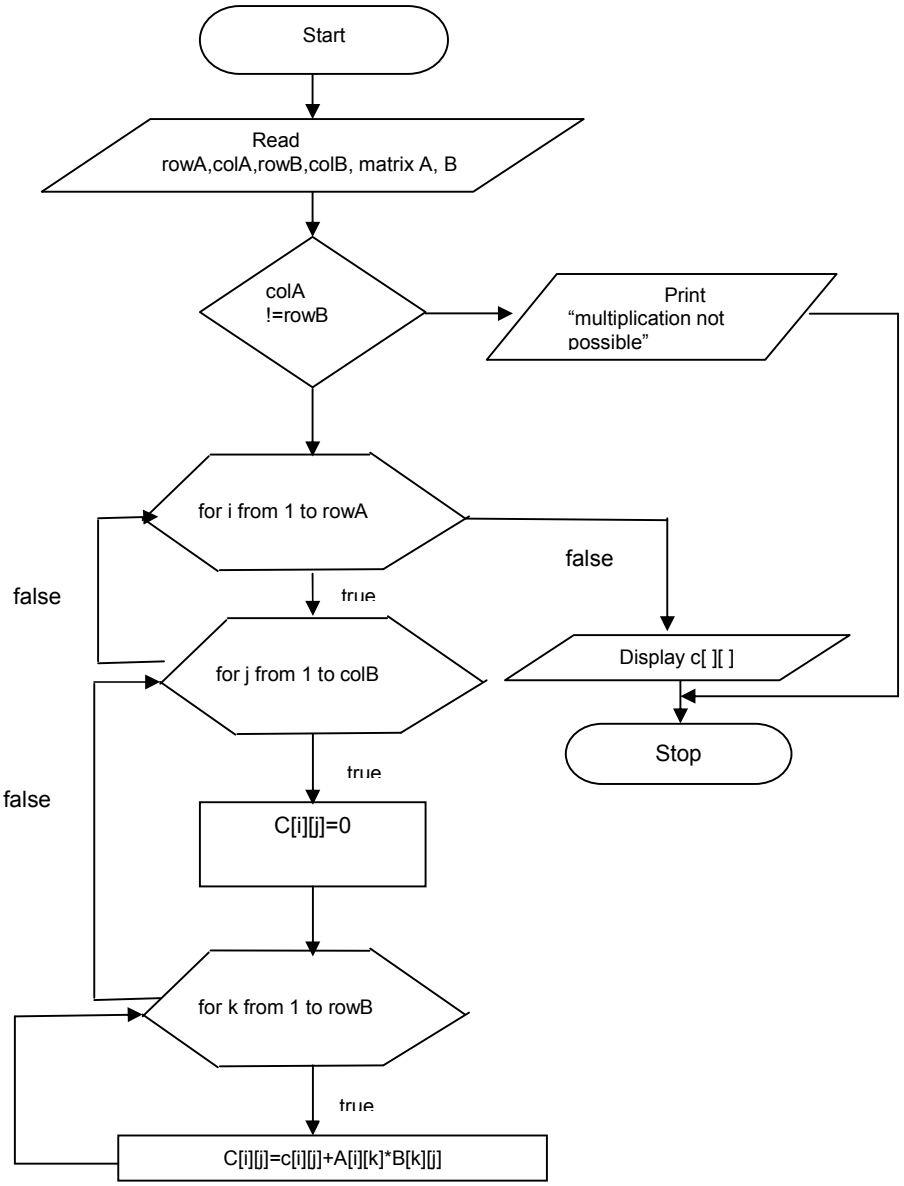
Algorithm

Step 1: start

Step 2: Read rows and columns of two matrices
Step 3: Read elements of first and second matrix to arrays A[ ][ ] and B[ ][ ]
Step 4: if columns [A] ≠ rows [B]
Then error "incompatible dimensions – Multiplication Not possible"
Repeat step 2 if required.

Step 5: For i varies from 1 to rows [A]
Step 6: For j varies from 1 to columns [B]
Step 7: initialize C[i,j]=0
Step 8: For k varies from 1 to columns [A]
Step 9: C[i, j]=C[i, j]+A[i, k]*B[k, j] , Repeat 5to 9 till the end of loops.
Step 10:Print result as C[ ][ ]
Step 11: Stop

Flowchart



Procedure:

Save program in a file, Compile program, debug errors, Execute or Run program with necessary inputs. Verify the outputs obtained.

Program:

```
/* Program to print product of two matrices*/
#include <stdio.h>
int main()
{
 int m, n, p, q, i, j, k, sum = 0;
 int first[10][10], second[10][10], multiply[10][10];

 printf("Enter the number of rows and columns of first matrix\n");
 scanf("%d%d", &m, &n);

 printf("Enter the number of rows and columns of second matrix\n");
 scanf("%d%d", &p, &q);

 if (n != p)
  printf("Matrices with entered orders can't be multiplied with each other.\n");
 else
 { printf("Enter the elements of first matrix\n");

  for (i = 0; i < m; i++)
   for (j = 0; j < n; j++)
    scanf("%d", &first[i][j]);

   printf("Enter the elements of second matrix\n");

   for (i = 0; i < p; i++)
    for (j = 0; j < q; j++)
     scanf("%d", &second[i][j]);

   for (i = 0; i < m; i++) {
    for (j = 0; j < q; j++) {
            multiply[i][j]=0;
     for (k = 0; k < p; k++) {
      multiply[i][j] = multiply[i][j] + first[i][k]*second[k][j];
     }

    }
   }

   printf("Product of entered matrices:-\n");

   for (i = 0; i < m; i++) {
    for (j = 0; j < q; j++)
     printf("%d\t", multiply[i][j]);

    printf("\n");
   }
 }

 return 0;
}
```

Input Output

Enter the number of rows and columns of first matrix 3 3

Enter the number of rows and columns of second matrix 3 3

Enter the elements of first matrix
1 2 4 5 2 1 4 5 2

the elements of second matrix
1 2 4 5 2 1 4 5 2

Product of entered matrices

10	18	28
50	18	7
40	45	14

Result: The program is compiled, executed and the output is verified.

Sample questions

1.Program to find each rowsum and columnsum of a matrix.
2.Program to find transpose of a given matrix
3.Program to find diagonal sum of a matrix
4.Program to find sum of two matrices

Experiment 9

Program to concatenate two strings without using library functions

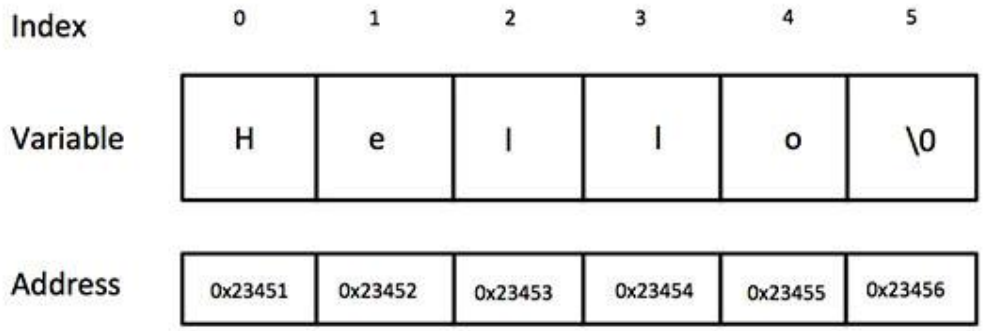
AIM: Program to print the concatenated string from two strings given.

Objectives: Study about string –using char arrays - in C language.

Tools/Equipents : Editors like gedit,vi editor in linux with gcc (or TurboC editor in windows) in desktop or laptop computer.

THEORY:

Strings are actually one-dimensional array of characters terminated by a **null** character '\0'



1 **strcpy(s1, s2);**

Copies string s2 into string s1.

2 **strcat(s1, s2);**

Concatenates string s2 onto the end of string s1.

3 **strlen(s1);**

Returns the length of string s1.

4 **strcmp(s1, s2);**

Returns 0 if s1 and s2 are the same; less than 0 if s1<s2; greater than 0 if s1>s2.

ALGORITHM:

step1: start

step2: Read two strings in two arrays str1,str2; let 1=j=0;

Step 3:if str1[i] not equal to '\0' ,increment i, repeat this step

Step 4:if str2[j] not equal to '\0' go to next step otherwise go to step 7

Step 5:str1 [i]=str2[j]

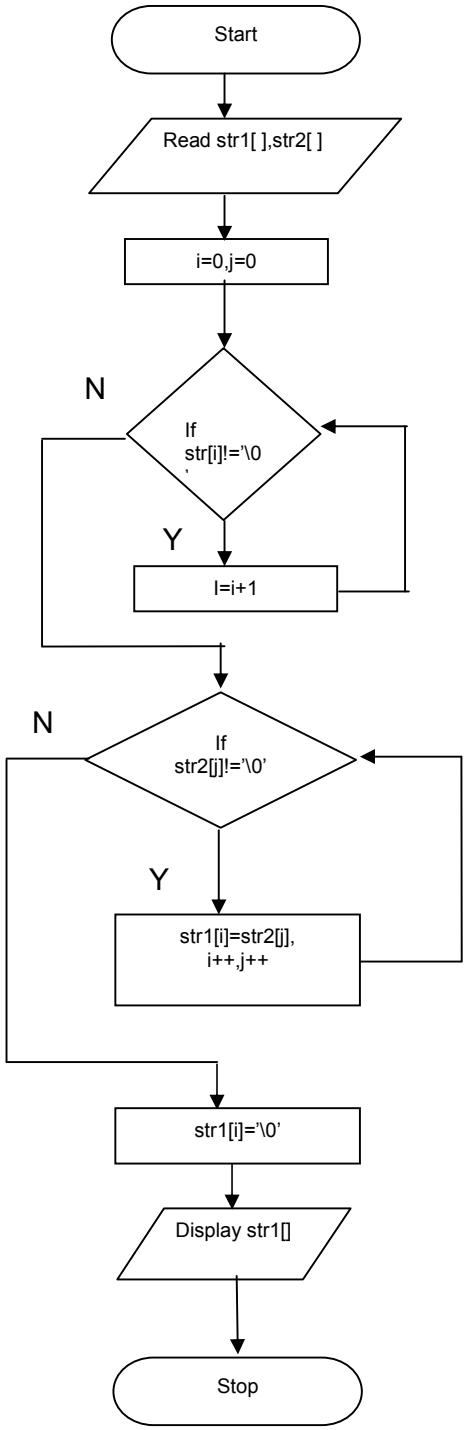
Step 6: increment i and j ,go to step 4

Step 7:str1 [i]=’\0’

Step 8: Print Concatenated string str1

step: stop

FLOWCHART



Procedure

Save program in a file, Compile program, debug errors, Execute or Run program with necessary inputs. Verify the outputs obtained.

PROGRAM

/*Program to concatenate two strings*/

```
#include<stdio.h>
void main(void)
{
 char str1[25],str2[25];
 int i=0,j=0;
 printf("\nEnter First String:");
 gets(str1);
 printf("\nEnter Second String:");
 gets(str2);
 while(str1[i]!='\0')
 i++;
 while(str2[j]!='\0')
 {
  str1[i]=str2[j];
  j++;
  i++;
 }
 str1[i]='\0';
 printf("\nConcatenated String is %s",str1);
}
```

Input Output

Enter the first string hello

Enter the second string world

Concatenated String is helloworld

Result: The program is compiled, executed and the output is verified

Sample questions

1. Program to sort a list of names alphabetically
2. Program to compare two strings and to print the characters that do not match.
3. Program to replace a substring with another string in a line of text.
4. Program to delete a substring from a line of text

Experiment 10

Program to print the elements of array using pointers

AIM: Program to print the elements of array using pointers

Objectives: Study about pointers in C language.

Tools/Equipents: Editors like gedit,vi editor in linux with gcc (or TurboC editor in windows) in desktop or laptop computer.

THEORY

A **pointer** is a variable whose value is the address of another variable, i.e., direct address of the memory location. Like any variable or constant, you must declare a pointer before using it to store any variable address. The general form of a pointer variable declaration is −

```
type *var-name;
```

Here, **type** is the pointer's base type; it must be a valid C data type and **var-name** is the name of the pointer variable. The asterisk * used to declare a pointer is the same asterisk used for multiplication. However, in this statement the asterisk is being used to designate a variable as a pointer.

Every variable is a memory location and every memory location has its address defined which can be accessed using ampersand (&) operator, which denotes an address in memory.

(a) We define a pointer variable, int *p
(b) assign the address of a variable to a pointer p=&x, x is an integer variable
(c) finally access the value at the address available in the pointer variable. v=*p

ALGORITHM:

step1: start

step2: Read array a[] with n elements

Step 3: initialize pointer p=&a[0] [or p=a]

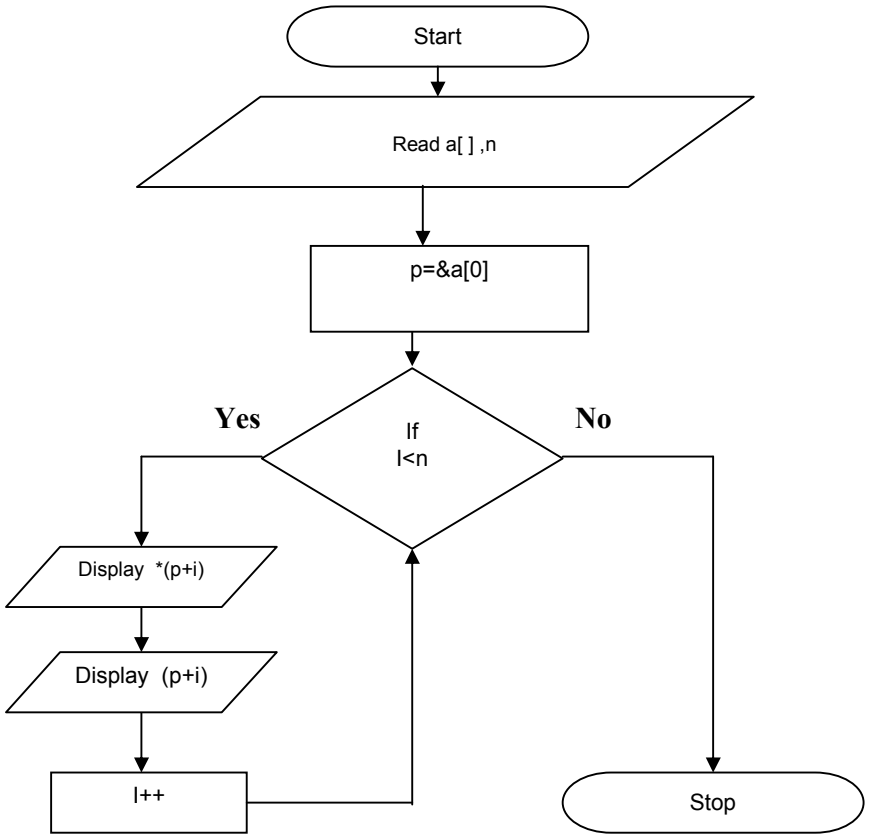
Step 4:if i<n go to next step otherwise go to step 7

Step 5: print *(p+i)

Step 6:i=i+1 go to step 4

Step 7: stop

FLOWCHART



Procedure

Save program in a file, Compile program, debug errors, Execute or Run program with necessary inputs. Verify the outputs obtained.

PROGRAM

/*Program to print the elements of array using pointers*

```
#include<stdio.h>
main()
{
int a[5]={5,4,6,8,9};
int *p=&a[0];
int i;
clrscr();
for(i=0;i<5;i++)
printf("%d ",*(p+i));
getch();
}
```

Input Output

5 4 6 8 9

Result: The program is compiled, executed and the output is verified

Sample questions

1. Program to find the maximum number in array using pointer.
2. Program to sort an array using pointers

Experiment 11

Program to find factorial of a given number using function.

Aim: Program to find factorial of a given number using function.

Objectives: Study about function operations in C language.

Tools/Equipents: Editors like gedit, vi editor in linux with gcc (or TurboC editor in windows) in desktop or laptop computer.

Theory

A function is a group of statements that together perform a task. Every C program has at least one function, which is **main()**, and all the most trivial programs can define additional functions.

You can divide up your code into separate functions. How you divide up your code among different functions is up to you, but logically the division is such that each function performs a specific task.

A function **declaration** tells the compiler about a function's name, return type, and parameters. A function **definition** provides the actual body of the function.

The C standard library provides numerous built-in functions that your program can call. For example, **strcat()** to concatenate two strings, **memcpy()** to copy one memory location to another location, and many more functions.

A function can also be referred as a method or a sub-routine or a procedure, etc.

Defining a Function

The general form of a function definition in C programming language is as follows:-

```
return_type function_name( parameter list ) {
   body of the function
}
```

A function definition in C programming consists of a *function header* and a *function body*. Here are all the parts of a function −

- **Return Type** − A function may return a value. The **return_type** is the data type of the value the function returns. Some functions perform the desired operations without returning a value. In this case, the return_type is the keyword **void**.
- **Function Name** − This is the actual name of the function. The function name and the parameter list together constitute the function signature.
- **Parameters** − A parameter is like a placeholder. When a function is invoked, you pass a value to the parameter. This value is referred to as actual parameter or argument. The parameters in function definition that receive these argument values are known as formal parameters. The parameter list refers to the type, order, and number of the parameters of a function. Parameters are optional; that is, a function may contain no parameters.

- **Function Body** − The function body contains a collection of statements that define what the function does.

Function Declarations

A function **declaration** tells the compiler about a function name and how to call the function. The actual body of the function can be defined separately.

A function declaration has the following parts:-

```
return_type function_name( parameter list );
```

Calling a Function

While creating a C function, you give a definition of what the function has to do. To use a function, you will have to call that function to perform the defined task.

To call a function, you simply need to pass the required parameters along with the function name, and if the function returns a value, then you can store the returned value.

Recursion is the process of repeating items in a self-similar way. In programming languages, if a program allows you to call a function inside the same function, then it is called a recursive call of the function.

```
void recursion() {
  recursion(); /* function calls itself */
}

int main() {
  recursion();
}
```

The C programming language supports recursion, i.e., a function to call itself. But while using recursion, programmers need to be careful to define an exit condition from the function, otherwise it will go into an infinite loop.

Recursive functions are very useful to solve many mathematical problems, such as calculating the factorial of a number, generating Fibonacci series, etc.

Number Factorial

The following example calculates the factorial of a given number using a recursive function:-

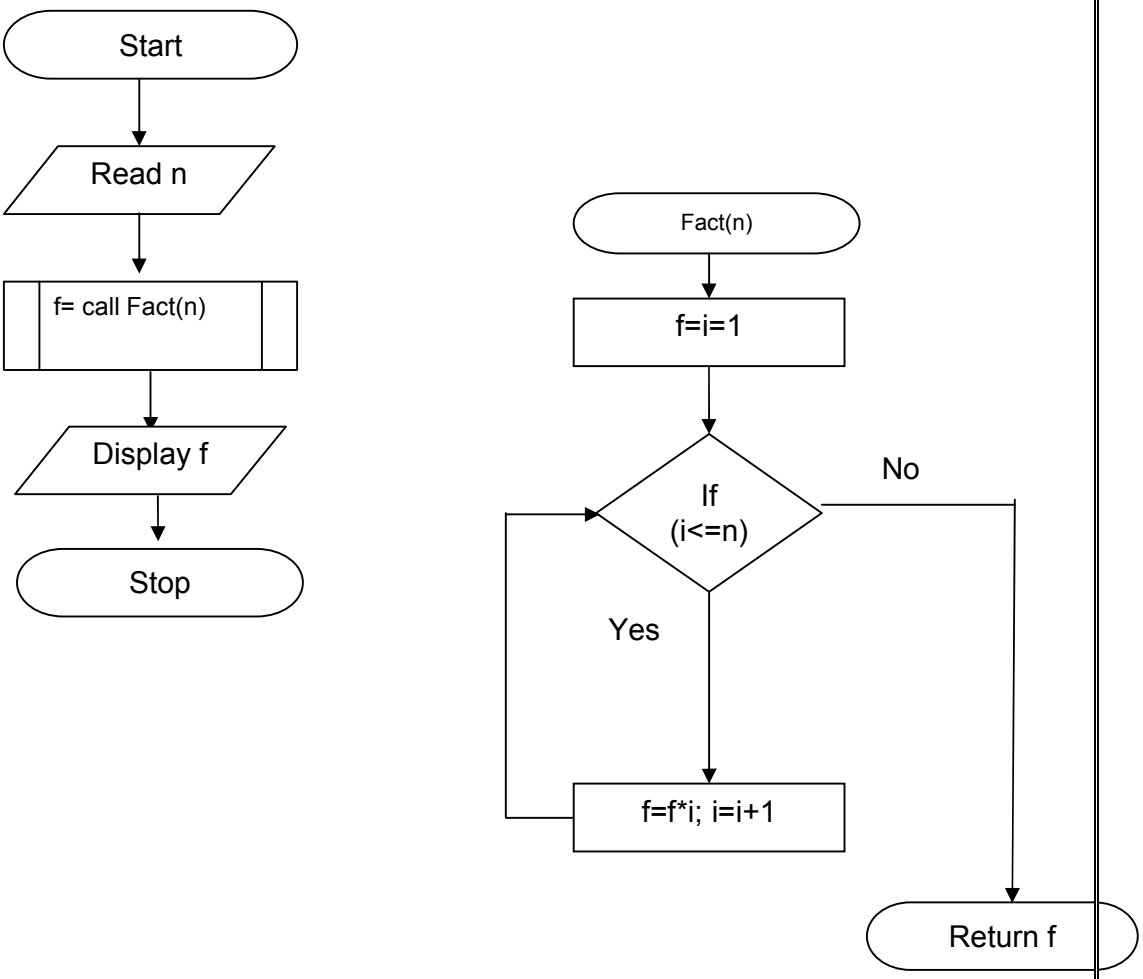
```
#include <stdio.h>
int factorial(unsigned int i) {

  if(i <= 1) {
    return 1;
  }
  return i * factorial(i - 1);
}
```

```
int  main() {
   int i = 15;
   printf("Factorial of %d is %d\n", i, factorial(i));
   return 0;
}
```

When the above code is compiled and executed, it produces the following result:-

Factorial of 15 is 2004310016

FLOWCHART



ALGORITHM

step 1. Start
step 2. Read the number n
step 3.call function f=fact(n)
step 4.print f
step 5. Stop

function fact[n]

step1: i=1, fact=1
step 2. If(i<=n)go to step 3 otherwise go to step 6
step 3. fact=fact*i
step 4. i=i+1
step 5: go to step 2
step 6: return fact

Procedure

Save program in a file, Compile program, debug errors, Execute or Run program with necessary inputs. Verify the outputs obtained.

PROGRAM

```
/*program to find factorial of a given number*/
#include<stdio.h>
#include<math.h>
void main()
{
int n;
long int f;
long int fact(int);

clrscr();
printf("enter a number");
scanf("%d",&n);
f=fact(n);
printf("\nfactorial of a given no is: %d ",f);
getch();
}
long int fact(int n)
{
int i;
long int f=1;
for(i=1;i<=n;i++)
{
        f=f*i;
}
return f;
}
```

Input Output

Enter a number 5
Factorial of a given no is: 120

Result: The program is compiled, executed and the output is verified

Sample questions

1. Write a program to find the value of
   $e^x=1+ x/1! +x^2/2! +……..+x^n/n!$
2. Write a program to find the value of
   $\sin x= x- x^3/3! +x^5/5! –x^7/7! +…….$
3. Program to insert an element at a relevant position in an array using function
4. Program to find the alphanumeric characters and their count in a line of text.
5. Program to find x^n
6. Write a program to display N fibonacci numbers using recursion.

Experiment 12

Program to find total mark of n students

Aim: Program to find total mark out of 5 subjects of n students

Objectives: Study about structure concepts in C language.

Tools/Equipents: Editors like gedit, vi editor in linux with gcc (or TurboC editor in windows) in desktop or laptop computer.

Theory

Structure is a user defined data type available in C that allows to combine data items of different types

To define a structure, you must use the **struct** statement. The struct statement defines a new data type, with more than one member. The format of the struct statement is as follows:-

```
struct [structure tag] {


   member definition;
   member definition;
   ...
   member definition;
} [one or more structure variables(optional)];
```

To access any member of a structure, we use the **member access operator (.)**.

The member access operator is coded as a period between the structure variable name and the structure member that we wish to access.

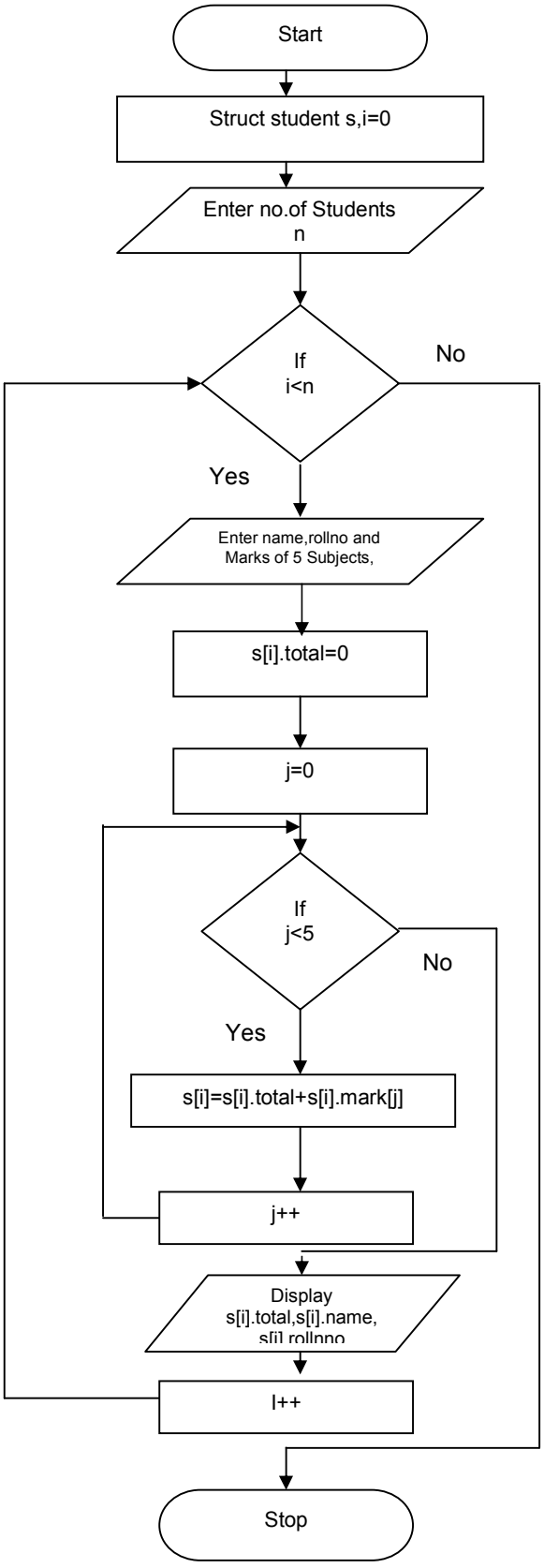
We would use the keyword **struct** to define variables of structure type.

```
struct Books Book1;

Book1.title
```

To access the members of a structure using a pointer to that structure, you must use the → operator as follows:-

```
struct Books *struct_pointer;
struct_pointer->title;
```

FLOWCHART


Start
Struct student s,i=0
Enter no.of Students
n
If
i<n
No
Yes
Enter name,rollno and
Marks of 5 Subjects,
s[i].total=0
j=0
If
j<5
No
Yes
s[i]=s[i].total+s[i].mark[j]
j++
Display
s[i].total,s[i].name,
s[i].rollnno
I++
Stop

Algorithm

Step 1: start

Step 2: Define user data type for student.

Step 3. Read number of students n

Step 4: for i varies from 0 to n-1 , repeat steps 5 to 9 otherwise goto step 10

Step 5: Read student struct variable, s[i].name,s[i].rollno

Step 6: for j varies from 0 to 4, repeat steps 7,8 otherwise go to step9

Step 7: read mark as s[i].mark[j]

Step8 :s[i].total = s[i].total+s[i].mark[j]

Step 9:display s[i].name,s[i].rollno, and s[i].total

Step 10:stop

Procedure

Save program in a file, Compile program, debug errors, Execute or Run program with necessary inputs. Verify the outputs obtained.

PROGRAM

```
/*A program to find the total marks*/
#include<stdio.h>
struct student
{
char name[10];
int rollno;
int subject[5],total;
};
main ( )
{
static struct student s[100];
int n,i,j;
clrscr();
```

```
printf("enter the no.of students");
scanf("%d",&n);

printf("enter the marks of five subjects");
for(i=0;i<n;i++)
{
printf(“enter student roll number \n”);
scanf(“%d”,&s[i].rollno);
printf(“enter student name \n”);
scanf(“%s”, s[i].name);

printf("enter s[%d] student marks\n",i);
s[i].total=0;
for(j=0;j<5;j++)
{
scanf("%d",&s[i].subject[j]);
s[i].total=s[i].total+s[i].subject[j];
}
printf("%d\t%s\t%d\n",s[i].rollno,s[i].name,s[i].total);
}
}
```

Input Output

```
enter the no.of students2
enter student roll number
1
enter student name
anu

enter the marks of five subjects
enter s[0] student marks
1
2
3
4
5
1        anu      15
enter student roll number
```

```
2
enter student name
akhil

enter s[1] student marks
12
32
14
15
65
2        akhil    138
```

Result: The program is compiled, executed and the output is verified.

Sample questions

1. Prepare salary bill of an employee with necessary details of salary.
2. Write a program to perform banking operations withdrawal and deposit in a bank.
3. Write a C program that accepts the following –Name,Regno,mark1,mark2 and mark3 of students. Display an alphabetical list of students with the following details-
   Name, Regno, Total marks and Rank in the class.