**Part -A**

**1.Develop a program to solve simple computational problems using arithmetic expressions and use of each operator leading to simulation of a commercial calculator.(No built-in math function).**

#include<stdio.h>

main()

{

float a,b;

char op;

float res;

printf("enter operands\n");

scanf("%f%f",&a,&b);

printf("enter operator\n");

scanf("%s",&op);

switch(op)

{

case '+':res=a+b;

printf("sum=%f\n",res);

break;

case '-':res=a-b;

printf("difference=%f\n",res);

break;

case '\*':res=a\*b;

printf("product=%f",res);

break;

case '/':if(b!=0)

{

res=a/b;

printf("quotient=%f",res);

}

else

{

printf("divide by zero error\n");

}

break;

default:printf("invalid error\n");

break;

}

}

**Execution**:

gedit cal.c

gcc cal.c –o cal.out

./cal.out

**Outputs**:

-----------------------------------------------------------------------------enter operands

5

3

enter operator

+

sum=8.000000

---------------------------------------------------------------------------------------------------------------------

enter operands

5

2

enter operator

-

difference=3.000000

---------------------------------------------------------------------------------------------------------------------

enter operands

9

8

enter operator

\*

product=72.000000

---------------------------------------------------------------------------------------------------------------------

enter operands

18

2

enter operator

/

quotient=9.000000

--------------------------------------------------------------------------------------------------------------------

enter operands

9

0

enter operator

/

divide by zero error

--------------------------------------------------------------------------------------------------------------------

enter operands

9

3

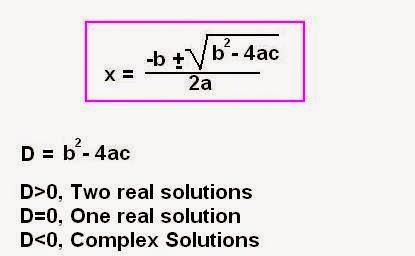
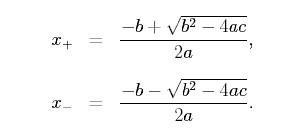
enter operator

|

invalid error

---------------------------------------------------------------------------------------------------------------------

**2.Develop a program to compute the roots of a quadratic equation by accepting the coefficients. Print appropriate messages**



#include<stdio.h>

#include<math.h>

int main()

{

float a, b, c, disc, root1, root2, real, imag;

printf("Enter values for a, b and c \n");

scanf("%f%f%f",&a,&b,&c);

if((a==0)&&(b==0))

{

printf("Roots cannot be determined\n");

}

else if(a==0)

{

printf("linear equation is\n");

root1=-c/b;

printf("root1=%.3f \n",root1);

}

else

{

disc=b\*b-4\*a\*c;

if(disc==0)

{

root1=root2=-b/(2\*a);

printf("The real and equal roots are :-\n");

printf("root1=root2=%.3f \n",root1);

}

else if(disc>0)

{

root1=(-b+sqrt(disc))/(2\*a);

root2=(-b-sqrt(disc))/(2\*a);

printf("The real and distinct roots are :-\n");

printf("root1=%.3f\troot2=%.3f \n",root1,root2);

}

else

{

real=-b/(2\*a);

imag=sqrt(fabs(disc))/(2\*a);

printf("The complex roots are :-\n");

printf("root1=%.3f+i%.3f\t", real, imag);

printf("root2=%.3f-i%.3f \n", real, imag);

}

}

return 0;

}

**Execution**:

gedit quad.c

gcc quad.c –o quad.out

./quad.out

**Outputs**:

--------------------------------------------------------------------------Enter values for a, b and c

0

0

0

Roots cannot be determined

----------------------------------------------------------------------------------------------------------------

Enter values for a, b and c

0

4

5

linear equation is root1=-1.250

----------------------------------------------------------------------------------------------------------------

Enter values for a, b and c

2

6

8

The complex roots are :-

root1=-1.500+i1.323

root2=-1.500-i1.323

---------------------------------------------------------------------------------------------------------------

Enter values for a, b and c

1

-4

3

The real and distinct roots are :-

root1=3.000

root2=1.000

----------------------------------------------------------------------------------------------------------------

Enter values for a, b and c

1

6

9

The real and equal roots are :-

root1=root2=-3.000

-------------------------------------------------------------------------------------------------------------------

**3.** **Develop a program to find the reverse of a positive integer and check for palindrome or not.Display appropriate messages.**

#include<stdio.h>

main()

{

int n,rev=0,rem,num;

printf("Enter the number \n");

scanf("%d",&num);

n=num;

while(num!=0)

{

rem=num%10;

num=num/10;

rev=rev\*10+rem;

}

printf("The reverse number is %d\n",rev);

if(rev==n)

printf("%d is a palindrome\n",n);

else

printf("%d is not palindrome\n",n);

return 0;

}

**Execution**:

gedit pal.c

gcc pal.c –o pal.out

./pal.out

**Outputs**:

-----------------------------------------------------------------------------

Enter the number

1221

The reverse no.is 1221

1221 is a palindrome

---------------------------------------------------------------------------------------------------------------------

Enter the number

1221

The reverse number is 1221

1221 is a palindrome

--------------------------------------------------------------------------------------------------------------------

Enter the number

19902

The reverse number is 20991

19902 is not palindrome

--------------------------------------------------------------------------------------------------------------------

Enter the number

1001

The reverse number is 1001

1001 is a palindrome

-------------------------------------------------------------------------------------------------------------------

Enter the number

-1001

The reverse number is -1001

-1001 is a palindrome

----------------------------------------------------------------------------------------------------------------

Enter the number

0101

The reverse number is 101

101 is a palindrome

----------------------------------------------------------------------------------------------------------------

**4.  An electricity board charges the following rates for the use of electricity: for the first 200   
units 80 paise per unit: for the next 100 units 90 paise per unit: beyond 300 units Rs 1 per unit.All users are charged a minimum of Rs.100 as meter charge. If the total amount is more than Rs. 400, then an additional surcharge of 15% of a total amount is charged.   
Write a program to read the name of the user, number of units consumed and print out the charges**.

#include<stdio.h>

main()

{

char name[50];

float units,charge;

printf("enter name\n");

scanf("%s",name);

printf("enter number of units consumed\n");

scanf("%f",&units);

charge=100;

if(units<=200)

{

charge=charge+(0.80\*units);

}

else if(units>200&&units<=300)

{

charge=charge+(0.80\*200)+(0.90\*(units-200));

}

else

{

charge=charge+(0.80\*200)+(0.90\*100)+(1\*(units-300));

}

if(charge>=400)

{

charge=charge+(charge\*0.15);

}

printf("total charge=%f",charge);

}

**Execution**:

gedit unit.c

gcc unit.c –o unit.out

./unit.out

**Outputs:**

**---------------------------------------------------------------------------------------------------------------------**

enter name

seetha

enter number of units consumed

150

total charge=220.000000

---------------------------------------------------------------------------------------------------------------------

enter name

amar

enter number of units consumed

250

total charge=305.000000

---------------------------------------------------------------------------------------------------------------------

enter name

amitha

enter number of units consumed

302

total charge=352.000000

---------------------------------------------------------------------------------------------------------------------

enter name

akshay

enter number of units consumed

350

total charge=460.000000

---------------------------------------------------------------------------------------------------------------------

**5. Introduce 1D Array manipulation and implement Binary Search.**

#include<stdio.h>

main()

{

int a[50],low,high,mid,found,n,key,i;

printf("enter no.of elements\n");

scanf("%d",&n);

printf("enter the elements\n");

for(i=0;i<=n-1;i++)

{

scanf("%d",&a[i]);

}

printf("enter elements to search");

scanf("%d",&key);

found=0;

low=0;

high=n-1;

while(low<=high)

{

mid=(low+high)/2;

if(key==a[mid])

{

found=1;

break;

}

if(key<a[mid])

{

high=mid-1;

}

if(key>a[mid])

{

low=mid+1;

}

}

if (found==1)

{

printf("search successful at position %d",mid+1);

}

else

{

printf("unsuccessful search\n");

}

}

**Exaecution** :

gedit binary.c

gcc binary.c -o binary.out

./binary.out

**Outputs** :

---------------------------------------------------------------------------------------------------------------------

enter no.of elements

5

enter the elements

5

3

4

1

7

enter elements to search

4

search successful at position 3

---------------------------------------------------------------------------------------------------------------------

enter no.of elements

5

enter the elements

5

3

4

1

7

enter elements to search

8

unsuccessful search

--------------------------------------------------------------------------------------------------------------------

**6. Implement using functions to check whether the given number is prime and display appropriate messages.(No built –in math function).**

#include<stdio.h>

int isprime(int num)

{

int i;

if(num==0||num==1)

{

return 0;

}

for(i=2;i<=num/2;i++)

{

if(num%i==0)

return (0);

}

return (1);

}

int main()

{

int n;

printf("enter the number");

scanf("%d",&n);

if(isprime(n))

{

printf("the given number %d is prime\n”,n);

}

else

{

printf(“the given number %d is not a prime\n”,n);

}

}

**Execution**:

gedit prime.c

gcc prime.c –o prime.out

./prime.out

**Outputs**:

---------------------------------------------------------------------------enter the number

7

the given number 7 is prime

---------------------------------------------------------------------------------------------------------------------

enter the number

16

the given number 7 is not a prime

---------------------------------------------------------------------------------------------------------------------

**Part B**

**7.Develop a program to introduce 2D Array manipulation and implement Matrix   
 multiplication and ensure the rules of multiplication are checked**.

#include<stdio.h>

int main()

{

int i,j,k,m,n,p,q,a[30][30],b[30][30],c[30][30];

printf("Enter the order of matrix A\n");

scanf("%d%d",&m,&n);

printf("Enter the order of matrix B\n");

scanf("%d%d",&p,&q);

if(n==p)

{

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

for(i=0;i<m;i++)

{

for(j=0;j<n;j++)

{

scanf("%d",&a[i][j]);

}

}

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

for(i=0;i<p;i++)

{

for(j=0;j<q;j++)

{

scanf("%d",&b[i][j]);

}

}

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]=c[i][j]+a[i][k]\*b[k][j];

}

}

}

printf("The product of two Matrix:\n");

for(i=0;i<m;i++)

{

for(j=0;j<q;j++)

{

printf("%d\t",c[i][j]);

}

printf("\n");

}

}

else

{

printf("\nMatrix multiplication is not possible\n");

}

return 0;

}

**Execution**:

gedit mat.c

gcc mat.c –o mat.out

./mat.out

**Outputs**:

------------------------------------------------------------------------------

Enter the order of matrix A

2 2

Enter the order of matrix B

2 2

Enter the elements of matrix A:

1 2

3 4

Enter the elements of matrix B:

5 6

7 8

The product of two Matrix:

19 22

43 50

---------------------------------------------------------------------------------------------------------------------

Enter the order of matrix A

2 3

Enter the order of matrix B

4 2

Matrix multiplication is not possible

---------------------------------------------------------------------------------------------------------------------

Enter the order of matrix A

3 3

Enter the order of matrix B

3 2

Enter the elements of matrix A:

1 2 3

3 4 5

2 1 2

Enter the elements of matrix B:

1 2

2 4

4 3

The product of two Matrix:

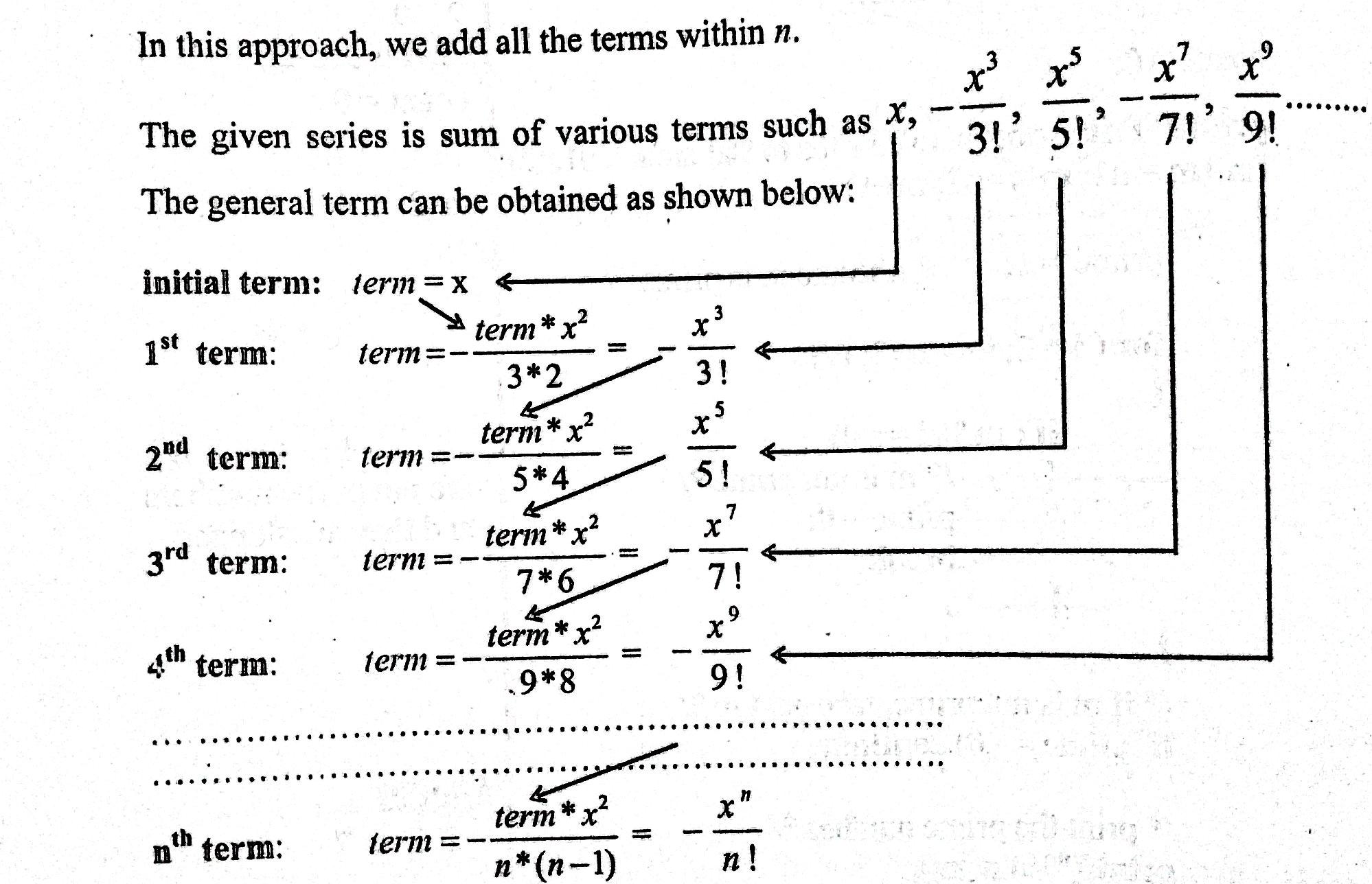
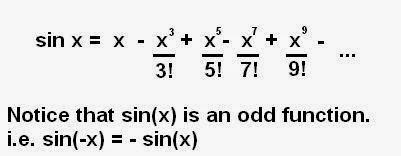
17 19

31 37

12 14

--------------------------------------------------------------------------------------------------------------------

**8. Develop a program to compute Sin(x) using Taylor series approximation. Compare your   
 result with built –in Library function. Print both the results with appropriate messages.**



#include<stdio.h>

#include<math.h>

main()

{

int i,n;

float x,sum,term;

printf("Enter the value for x:");

scanf("%f", &x);

printf("Enetr no of terms:");

scanf("%d", &n);

x=x\*3.14159/180; //converting into degree

printf("The Mathematical Sin value is =%f\n", sin(x));

term=x;

sum=x;

for(i=1;i<n;i++)

{

term= - term\*x\*x/(2\*i\*(2\*i+1));

sum=sum+term;

}

printf("The Calculated Sin value is=%f\n", sum);

}

**Execution**:

gedit sine.c

gcc sine.c –o sine.out

./sine.out

**Outputs**:

------------------------------------------------------------------------------

Enter the value for x:0

Enetr no of terms:0

The Mathematical Sin value is =0.000000

The Calculated Sin value is=0.000000

---------------------------------------------------------------------------------------------------------------------

Enter the value for x:12

Enetr no of terms:5

The Mathematical Sin value is =0.207912

The Calculated Sin value is=0.207912

---------------------------------------------------------------------------------------------------------------------

**9.Write fuctions to implement string operations such as compare, concatenate, string   
 length. Convince the parameter passing techniques**.

#include<stdio.h>

#include<string.h>

void slen(char s1[100])

{

int i=0,count=0;

while(s1[i]!='\0')

{

count++;

i++;

}

printf("length of string is %d\n",count);

}

void scat(char s1[100],char s2[100])

{

int i,j;

i=strlen(s1);

for(j=0;s2[j]!='\0';i++,j++)

{

s1[i]=s2[j];

}

s1[i]='\0';

printf("concatenated string is %s",s1);

}

void scompare(char s1[100],char s2[100])

{

int flag,i;

for(i=0;s1[i]!='\0'||s2[i]!='\0';i++)

{

if(s1[i]==s2[i])

{

flag=1;

}

else

{

flag=0;

break;

}

}

if(flag==1)

{

printf("strings are same\n");

}

else

{

printf("strings are not same\n");

}

}

void main()

{

char s1[100],s2[100];

printf("enter string1\n");

gets(s1);

printf("enter string2\n");

gets(s2);

slen(s1);

slen(s2);

scompare(s1,s2);

scat(s1,s2);

}

**Execution**:

gedit string.c

gcc string.c –o string.out

./string.out

**Outputs**:

------------------------------------------------------------------------------

enter string1

c programming

enter string2

laboratory

length of string is 13

length of string is 10

strings are not same

concatenated string is c programming laboratory

---------------------------------------------------------------------------------------------------------------------

enter string1

programming

enter string2

programming

length of string is 11

length of string is 11

strings are same

concatenated string is programmingprogramming

---------------------------------------------------------------------------------------------------------------------

**10.Develop a program to sort the given set of N numbers using Bubble sort.**

#include<stdio.h>

main()

{

int n,i,j,temp,a[100];

printf("Enter the value for n\n");

scanf("%d",&n);

printf("Enter %d elements into array\n",n);

for(i=0;i<n;i++)

{

scanf("%d",&a[i]);

}

printf("The unsorted array is :-\n");

for(i=0;i<n;i++)

{

printf("%d\t",a[i]);

}

for(i=1;i<n;i++)

{

for(j=0;j<(n-i);j++)

{

if(a[j]>a[j+1])

{

temp=a[j];

a[j]=a[j+1];

a[j+1]=temp;

}

}

}

printf("\nThe sorted array is :-\n");

for(i=0;i<n;i++)

{

printf("%d\t",a[i]);

}

}

**Execution**:

gedit bubble.c

gcc bubble.c –o bubble.out

./bubble.out

**Outputs**:

------------------------------------------------------------------------------

Enter the value for n

6

Enter 6 elements into array

12

34

56

23

21

7

The unsorted array is :-

12 34 56 23 21 7

The sorted array is :-

7 12 21 23 34 56

---------------------------------------------------------------------------------------------------------------------

**11. Develop a program to find the square root of a given number N and execute for all   
 possible inputs with appropriate messages. Note : Don’t use library function sqrt(n).**

#include<stdio.h>

main()

{

int i;

float x,n;

printf("Enter a positive integer");

scanf("%f",&n);

x=n;

for(i=0;i<10;i++)

{

x=(x\*x+n)/(2\*x);

}

printf("The calculated square root is=%f\n",x);

}

**Execution**:

gedit sqrt.c

gcc sqrt.c –o sqrt.out

./sqrt.out

**Outputs**:

---------------------------------------------------------------------------------------------------------------------

Enter a positive integer

1

The calculated square root is=1.000000

---------------------------------------------------------------------------------------------------------------------

Enter a positive integer

4

The calculated square root is=2.000000

---------------------------------------------------------------------------------------------------------------------

Enter a positive integer

9

The calculated square root is=3.000000

---------------------------------------------------------------------------------------------------------------------

Enter a positive integer

90

The calculated square root is=9.486833

---------------------------------------------------------------------------------------------------------------------

**12. Implement structures to read, write and compute average-marks and the students scoring above and below the average marks for a class of N students.**

#include<stdio.h>

struct student

{

char name[50];

int marks;

};

void main()

{

struct student s[100];

int n,i;

float average=0,sum=0;

printf("enter number of students\n");

scanf("%d",&n);

printf("enter student details\n");

for(i=0;i<=n-1;i++)

{

printf("enter student name\n");

scanf("%s",s[i].name);

printf("enter marks\n");

scanf("%d",&s[i].marks);

}

for(i=0;i<=n-1;i++)

{

sum=sum+s[i].marks;

}

average=sum/n;

printf("average marks=%f",average);

printf("students who scored above average are\n");

for(i=0;i<=n-1;i++)

{

if(s[i].marks>average)

{

printf("%s\t%d\n",s[i].name,s[i].marks);

}

}

printf("students who scored below average are\n");

for(i=0;i<=n-1;i++)

{

if(s[i].marks<average)

{

printf("%s\t%d\n",s[i].name,s[i].marks);

}

}

}

**Execution**:

gedit structure.c

gcc structure.c –o structure.out

./structure.out

**Outputs**:

---------------------------------------------------------------------------------------------------------------------

enter number of students

3

enter student details

enter student name

amar

enter marks

90

enter student name

ananya

enter marks

35

enter student name

likith

enter marks

50

average marks=58.333332students who scored above average are

amar 90

students who scored below average are

ananya 35

likith 50

---------------------------------------------------------------------------------------------------------------------

enter number of students

5

enter student details

enter student name

shan

enter marks

82

enter student name

smith

enter marks

42

enter student name

navya

enter marks

51

enter student name

vijay

enter marks

94

enter student name

sanjay

enter marks

21

average marks=58.000000students who scored above average are

shan 82

vijay 94

students who scored below average are

smith 42

navya 51

sanjay 21

---------------------------------------------------------------------------------------------------------------------

**13.Develop a program to using pointers to compute the sum, mean and the standard   
 deviation of all elements stored in an array of n real numbers.**

#include <stdio.h>

#include <math.h>

void main()

{

int i, n;

float x[10],\*p,mean, variance, sd, sum = 0, sum1 = 0;

printf("Enter the value of N \n");

scanf("%d", &n);

printf("Enter %d array elements \n", n);

for (i = 0; i < n; i++)

{

scanf("%f", &x[i]);

}

**/\* Compute the sum of all elements \*/**

p=x;

for (i = 0; i < n; i++)

{

sum = sum + \*p;

p++;

}

mean = sum / n;

**/\* Compute variance and standard deviation \*/**

p=x;

for (i = 0; i < n; i++)

{

sum1 = sum1 + pow((\*p - mean), 2);

p++;

}

variance = sum1 /n;

sd= sqrt(variance);

printf("SUM = %**.**2f\n", sum);

printf("MEAN = %**.**2f\n", mean);

printf("VARIANCE= %**.**2f\n", variance);

printf("STANDARD DEVIATION = %**.**2f\n", sd);

}

**Execution**:

gedit pointer.c

gcc pointer.c –o pointer.out

./pointer.out

**Outputs**:

---------------------------------------------------------------------------------------------------------------------

Enter the value of N

5

Enter 5 array elements

1

2

3

4

5

SUM = 15.00

MEAN = 3.00

VARIANCE= 2.00

STANDARD DEVIATION = 1.41

---------------------------------------------------------------------------------------------------------------------

Enter the value of N

6

Enter 6 array elements

12

13

44

34

44

32

SUM = 179.00

MEAN = 29.83

VARIANCE= 170.81

STANDARD DEVIATION = 13.07

---------------------------------------------------------------------------------------------------------------------

**14. Implement Recursive functions for Binary to Decimal Conversion.**

#include<stdio.h>

int main()

{

int bin,res;

printf("enter the number in binary format\n");

scanf("%d",&bin);

res=btod(bin);

printf("decimal equivalent=%d",res);

}

int btod(int bin)

{

if(bin==0)

{

return 0;

}

else

{

return(bin%10+2\*btod(bin/10));

}

}

**Execution**:

gedit binary.c

gcc binary.c –o binary.out

./binary.out

**Outputs**:

---------------------------------------------------------------------------------------------------------------------

enter the number in binary format

0001

decimal equivalent=1

---------------------------------------------------------------------------------------------------------------------

enter the number in binary format

1010

decimal equivalent=10

---------------------------------------------------------------------------------------------------------------------

enter the number in binary format

11010

decimal equivalent=26

---------------------------------------------------------------------------------------------------------------------