

|  |
| --- |
| CPNM LAB REPORT |
|  |
| April 2  BCSE FIRST YEAR FIRST SEMESTER  Authored by: SOHAM CHOWDHURY |



**CPNM LAB ASSIGNMENT REPORT**

BCSE FIRST YEAR FIRST SEMESTER 2021-2022

NAME-SOHAM CHOWDHURY

DEPARTMENT-COMPUTER SCIENCE AND ENGINEERING

SECTION-A3.

ROLL NO-002110501145.

# ASSIGNMENT 8

1. Write a menu-driven program for solving a system of linear equations using Gauss-Elimination method, Jacobi’s method and Gauss Elimination with pivoting method

PROGRAM:

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

#define e 0.001

void input(double a[20][20],int n)

{

    printf("enter the augmented matrix(coefficients):-\n");

    int i,j;

    for(i=0;i<n;i++)

    {

        printf("enter the coefficients of the equation number %d\n",i+1);

        for(j=0;j<n+1;j++)

        {

            scanf("%lf",&a[i][j]);

        }

    }

}

void print\_augmented\_matrix(double a[20][20],int n)

{

    printf("\nthe matrix is:-\n");

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n+1;j++)

        {

            if(j==n)

            printf(":%.0lf\t",a[i][j]);

            else

            printf("%.0lf\t",a[i][j]);

        }

        printf("\n");

    }

}

void guass\_elimination(double a[20][20],int n)

{

    double temp;

    int i,j,k;

    double x[10];

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            if(j>i)

            {

                temp=a[j][i]/a[i][i];

                for(k=0;k<n+1;k++)

                {

                    a[j][k]=a[j][k]-(temp\*a[i][k]);

                }

            }

        }

    }

   x[n-1]=a[n-1][n]/a[n-1][n-1];

   for(i=n-2;i>=0;i--)

   {

       double sum=0.0;

       for(j=i+1;j<=n;j++)

       {

           sum=sum+a[i][j]\*x[j];

       }

       x[i]=(a[i][n]-sum)/a[i][i];

   }

   printf("the solution is:\n");

   for(i=0;i<n;i++)

   printf("%0.3lf\t",x[i]);

}

void jacobis\_method(double a[20][20],int n)

{

    int m,c=1,j,i,k;

    printf("enter the maximum number of iterations=");

    scanf("%d",&m);

    double x\_old[100];

    double x\_new[100];

    double sum=0.0;

    for(i=0;i<n;i++)

    x\_old[i]=0;

    for(k=0;k<m;k++)

    {

        for(i=0;i<n;i++)

        {

            sum=0;

            for(j=0;j<n;j++)

            {

                if(i!=j)

                sum=sum+(a[i][j])\*x\_old[j];

            }

            x\_new[i]=(a[i][n]-sum)/a[i][i];

        }

        for(i=0;i<n;i++)

        {

            if(fabs(x\_old[i]-x\_new[i])<e)

            {

                c=0;

            }

            else

            c=1;

        }

        if(c==1)

        {

            for(j=0;j<n;j++)

            {

                x\_old[j]=x\_new[j];

            }

        }

        else

        break;

    }

    printf("the solutions are:\n");

    for(k=0;k<n;k++)

    printf("%.3lf\t",x\_new[k]);

}

void pivoting\_method(double a[20][20],int n)

{

    double temp;

    int i,j,k;

    double x[10];

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n+1;j++)

        {

            if(j!=i)

            {

                temp=a[j][i]/a[i][i];

                for(k=0;k<n+1;k++)

                {

                    a[j][k]=a[j][k]-(temp\*a[i][k]);

                }

            }

        }

    }

    for(i=0;i<n;i++)

    {

        x[i]=a[i][n]/a[i][i];

    }

    printf("the solutions are as follows:-\n");

    for(i=0;i<n;i++)

    {

        printf("%0.3lf\t",x[i]);

    }

}

int main()

{

    int n,choice;

    double arr[20][20];

    printf("the number of equations=");

    scanf("%d",&n);

    input(arr,n);

    print\_augmented\_matrix(arr,n);

    printf("enter the choice:-\n1.GUASS ELIMINATION\n2.JACOBI METHOD\n3.PIVOTING METHOD\n=");

    scanf("%d",&choice);

    switch(choice)

    {

        case 1:

        guass\_elimination(arr,n);

        print\_augmented\_matrix(arr,n);

        break;

        case 2:

        jacobis\_method(arr,n);

        break;

        case 3:

        pivoting\_method(arr,n);

        break;

        default:

        printf("invalid choice");

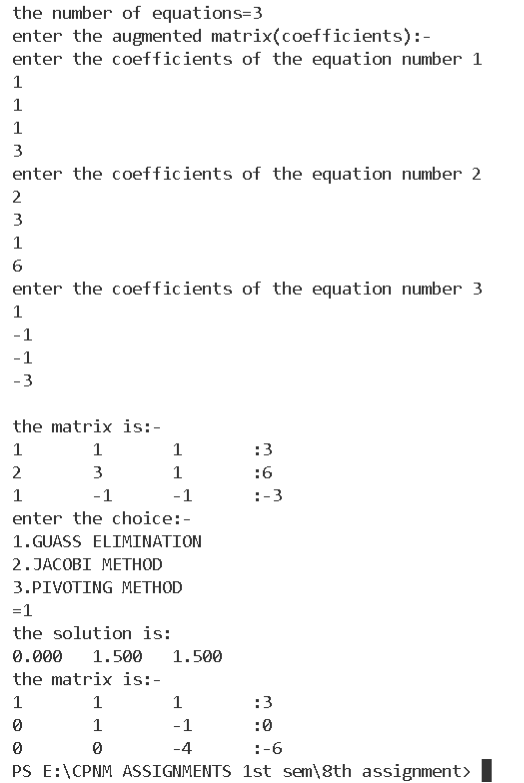
    }

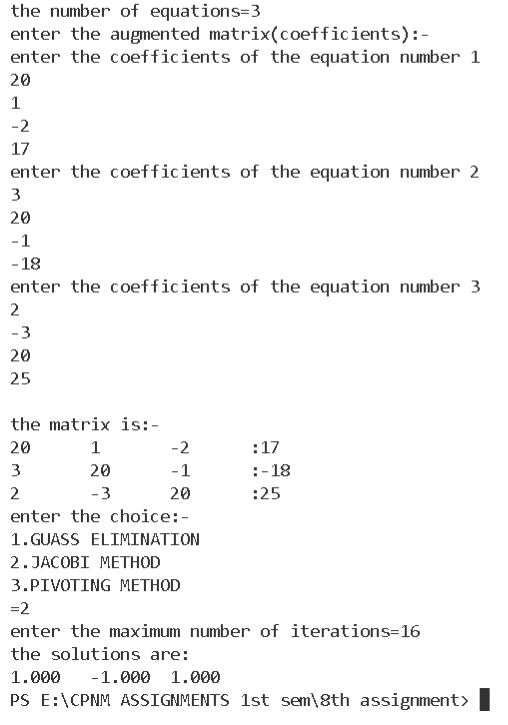
    return 0;

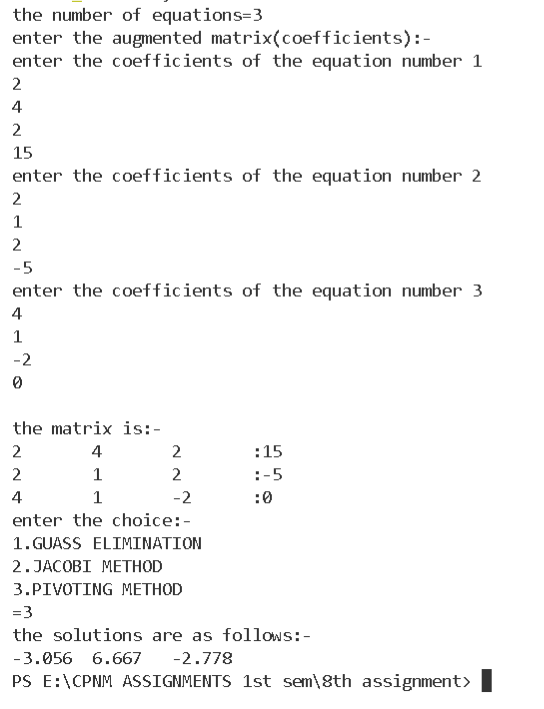
}

2. Using the above program solve the following system of equations: i. 𝑥 + 𝑦 + 𝑧 = 6 ii. 𝑥1 + 𝑥2 + 𝑥3 = 3 iii. 2𝑥1 + 4𝑥2 + 2𝑥3 = 15 𝑥 + 𝑦 − 𝑧 = 0 2𝑥1 + 3𝑥2 + 𝑥3 = 6 2𝑥1 + 𝑥2 + 2𝑥3 = −5 𝑥 − 𝑦 + 𝑧 = 2 𝑥1 − 𝑥2 − 𝑥3 = −3 4𝑥1 + 𝑥2 − 2𝑥3 = 0

OUTPUT:







3.Write a menu-driven program for implementing Interpolation using Lagrange's formula, Newton’s forward difference formula, and Newton’s backward difference formula.

PROGRAM:

#include<stdio.h>

#include<math.h>

#include<stdlib.h>

void input(float f[10][10],int n)

{

    printf("enter the values of x and f(x) respectively\n");

    fflush(stdin);

    for(int i=0;i<n;i++)

    {

        printf("value of x%d=",i+1);

        scanf("%f",&f[i][0]);

        printf("value of f(x%d)=",i+1);

        scanf("%f",&f[i][1]);

    }

}

void show\_input(float f[10][10],int n)

{

    printf("the inputs are as follows:-\n x\tf(x)\n");

    for(int i=0;i<n;i++)

    {

        printf("%.3f\t%.3f\n",f[i][0],f[i][1]);

    }

}

int factorial(int n)

{

    if((n==0)||(n==1))

    return 1;

    else

    return (n\*factorial(n-1));

}

double compute\_val(float f[10][10],int n,int c,float u)

{

    double sum=0;

    double product=1.00;

    float a=u;

    for(int j=1;j<=n;j++)

    {

        if(c==1)

        {

            for(int i=0;i<j-1;i++)

            {

                product=product\*a;

                a--;

            }

            sum=sum+((product\*f[0][j])/factorial(j-1));

        }

        else

        {

            for(int i=0;i<j-1;i++)

            {

                product=product\*a;

                a++;

            }

            sum=sum+((product\*f[n-1][j])/factorial(j-1));

        }

        product=1;

        a=u;

    }

    return sum;

}

void newtons\_forward\_difference(float f[10][10],int n,float x)

{

    float diff,a,h,u;

    double sum=0;

    int k=n;

    for(int j=1;j<=n;j++)

    {

        for(int i=0;i<k-1;i++)

        {

            diff=f[i+1][j]-f[i][j];

            f[i][j+1]=diff;

            diff=0;

        }

        k--;

    }

    h=f[1][0]-f[0][0];

    a=f[0][0];

    u=(x-a)/h;

    printf("f(%f)=%lf",x,compute\_val(f,n,1,u));

}

void newtons\_backward\_difference(float f[10][10],int n,float x)

{

    float diff,a,h,u;

    double sum=0;

    for(int j=1;j<=n;j++)

    {

        for(int i=n-1;i>j-1;i--)

        {

            diff=f[i][j]-f[i-1][j];

            f[i][j+1]=diff;

            diff=0;

        }

    }

    a=f[n-1][0];

    h=f[n-1][0]-f[n-2][0];

    u=(x-a)/h;

    printf("f(%f)=%lf",x,compute\_val(f,n,0,u));

}

void lagranges\_formula(float f[10][10],int n,float x)

{

    double numerator,denominator,sum=0;

    for(int i=0;i<n;i++)

    {

        numerator=f[i][1];

        denominator=1;

        for(int j=0;j<n;j++)

        {

            if(i!=j)

            {

                numerator\*=(x-f[j][0]);

                denominator\*=(f[i][0]-f[j][0]);

            }

        }

        sum=sum+numerator/denominator;

    }

    printf("the value of f(%f)=%lf",x,sum);

}

int main()

{

    int choice,n;

    float f[10][10],x;

    fflush(stdin);

    printf("enter the number of entries=");

    scanf("%d",&n);

    printf("\n1.newtons forward difference formula\n2.newtons backward difference formula\n3.lagranges formula\nenter the choice=");

    fflush(stdin);

    scanf("%d",&choice);

    printf("enter the inputs:-\n");

    fflush(stdin);

    input(f,n);

    printf("enter the value at which you want the function value=");

    scanf("%f",&x);

    show\_input(f,n);

    switch(choice)

    {

        case 1:

        newtons\_forward\_difference(f,n,x);

        break;

        case 2:

        newtons\_backward\_difference(f,n,x);

        break;

        case 3:

        lagranges\_formula(f,n,x);

        break;

        default:

        printf("invalid choice");

        break;

    }

    return 0;

}

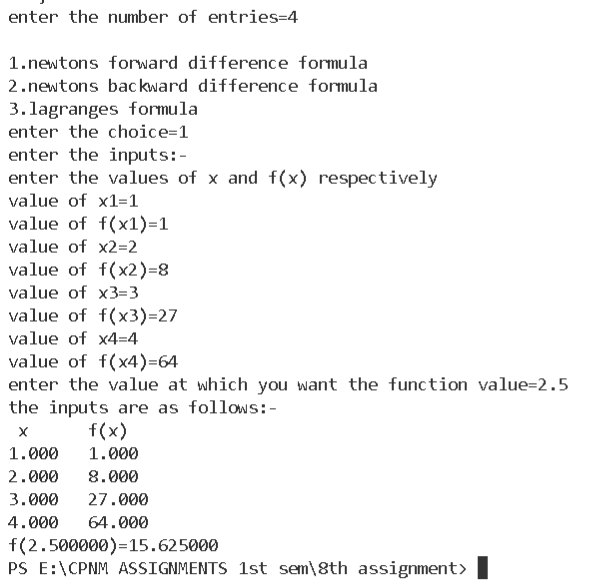
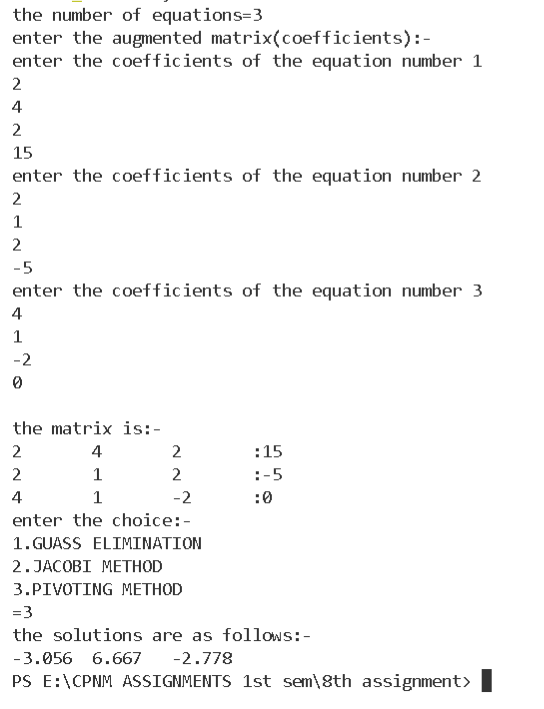
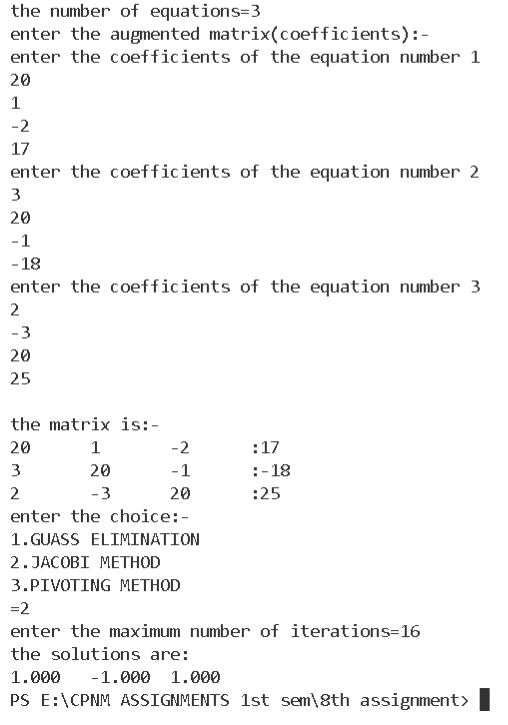
4. For the following table of values:

X 1 2 3 4

f(x) 1 8 27 64

Find f(2.5) using all three methods and comment on your answer

OUTPUT:



5. An experiment gave the following table of values for the dependent variable y for a set of known values of x. Obtain an appropriate least squares fit for the data.

x 1 2 3 4 5 6 7 8 9

y 5.5 7.0 9.6 11.5 12.6 14.4 17.6 19.5 20.5

PROGRAM:

#include<math.h>

#include<stdlib.h>

#include<stdio.h>

void input(float f[10][2],int n)

{

    printf("enter the values of x and f(x) respectively:-\n");

    for(int i=0;i<n;i++)

    {

        printf("x%d=",i+1);

        scanf("%f",&f[i][0]);

        printf("f(x%d)=",i+1);

        scanf("%f",&f[i][1]);

    }

}

void print(float f[10][2],int n)

{

    printf("the given data are as follows:-\n");

    printf("x\tf(x)\n");

    for(int i=0;i<n;i++)

    {

        printf("%f\t%f\n",f[i][0],f[i][1]);

    }

}

void curve\_fit(float f[10][2],int n)

{

    float sum\_x=0,sum\_x2=0,sum\_xy=0,sum\_y=0,a,b;

    for(int i=0;i<n;i++)

    {

        sum\_x+=f[i][0];

        sum\_x2+=pow(f[i][0],2);

        sum\_y+=f[i][1];

        sum\_xy+=f[i][0]\*f[i][1];

    }

    b=(n\*sum\_xy-sum\_x\*sum\_y)/(n\*sum\_x2-sum\_x\*sum\_x);

    a=(sum\_y - b\*sum\_x)/n;

    printf("the equation is :-y=%0.2f+%0.2fx",a,b);

}

int main()

{

    int n;

    printf("enter the number of entries=");

    scanf("%d",&n);

    float f[10][2];

    input(f,n);

    print(f,n);

    curve\_fit(f,n);

    return 0;

}

OUTPUT:

