ANNEXURE-I

Compilation of nonlinear partial differential equation for the gas flow in the matrix using JAVA.

```
import java.io.*;
import java.lang.Math;
import java.util.Scanner;
import java.util.HashSet;
import javax.swing.JFrame;
import javax.swing.SwingUtilities;
import java.io.File;
import java.io.IOException;
import jxl.*;
import jxl.read.biff.BiffException;
import jxl.write.*;
import jxl.write.Number;
import jxl.write.biff.RowsExceededException;
class Matrix
{
   public static double[][] invert(double a[][])
   {
      int n = a.length;
      double x[][] = new double[n][n];
      double b[][] = new double[n][n];
      int index[] = new int[n];
      for (int i=0; i<n; ++i)
         b[i][i] = 1;


      // Transform the matrix into an upper triangle
      gaussian(a, index);

      // Update the matrix b[i][j] with the ratios stored
      for (int i=0; i<n-1; ++i)
         for (int j=i+1; j<n; ++j)
            for (int k=0; k<n; ++k)
               b[index[j]][k]
               -= a[index[j]][i]*b[index[i]][k];

      // Perform backward substitutions
      for (int i=0; i<n; ++i)
      {
         x[n-1][i] = b[index[n-1]][i]/a[index[n-1]][n-1];
         for (int j=n-2; j>=0; --j)
         {
            x[j][i] = b[index[j]][i];
            for (int k=j+1; k<n; ++k)
            {
```

```
            x[j][i] -= a[index[j]][k]*x[k][i];
         }
         x[j][i] /= a[index[j]][j];
      }
   }
   return x;
}

// Method to carry out the partial-pivoting Gaussian
// elimination.  Here index[] stores pivoting order.

public static void gaussian(double a[][], int index[])
{
   int n = index.length;
   double c[] = new double[n];

   // Initialize the index
   for (int i=0; i<n; ++i)
      index[i] = i;

   // Find the rescaling factors, one from each row
   for (int i=0; i<n; ++i)
   {
      double c1 = 0;
      for (int j=0; j<n; ++j)
      {
         double c0 = Math.abs(a[i][j]);
         if (c0 > c1) c1 = c0;
      }
      c[i] = c1;
   }

   // Search the pivoting element from each column
   int k = 0;
   for (int j=0; j<n-1; ++j)
   {
      double pi1 = 0;
      for (int i=j; i<n; ++i)
      {
         double pi0 = Math.abs(a[index[i]][j]);
         pi0 /= c[index[i]];
         if (pi0 > pi1)
         {
            pi1 = pi0;
            k = i;
         }
```

```
        }

        // Interchange rows according to the pivoting order
        int itmp = index[j];
        index[j] = index[k];
        index[k] = itmp;
        for (int i=j+1; i<n; ++i)
        {
           double pj = a[index[i]][j]/a[index[j]][j];

           // Record pivoting ratios below the diagonal
           a[index[i]][j] = pj;

   }

   public double Tpc(double ych4, double yco2, double yn2, double yh2s)throws IOException,
BiffException, RowsExceededException, WriteException   //fun5
   {
      double temp = Pg(ych4, yco2, yn2, yh2s);
      return(168+(325*temp)-(12.5*temp*temp));
   }

   public double Ppr(double pm, double ych4, double yco2, double yn2, double yh2s) throws
IOException, BiffException, RowsExceededException, WriteException   //fun6
   {
      double temp = Ppc(ych4, yco2, yn2, yh2s);
      return(pm/temp);
   }

   public double Z(double pm, double ych4, double yco2, double yn2, double yh2s)throws
IOException, BiffException, RowsExceededException, WriteException    //fun7
   {
      double temp = Ppr(pm, ych4, yco2, yn2, yh2s);
      double A = 0.579;
      double B = 1.9894;
      double C = 0.0453;
      double D = 1.0528;

      return(A + ((1-A)/Math.exp(B)) + (C*(Math.pow(temp,D))));
   }

   public double Km(double pm)                        //fun8
   {
      double bk = 200;
      double kd = Math.pow((bk/12.639),(-1/0.33));
      return( kd * (1 + (bk/pm)));
```

```
    }

    public double Cm(double pm)                    //fun 9
    {
       return( 1/pm);
    }

    public double Va(double pm, double vl, double pl)                    //fun 10
    {
       return((vl*pm)/(pl+pm));
    }

    public double Vd(double pm,double vl, double pl)                    //fun 11
    {
       return( (vl*pl)/(pl+pm));
    }

    public double Tpr(double t, double ych4, double yco2, double yn2, double yh2s)throws
IOException, BiffException, RowsExceededException, WriteException          //fun12
    {
       double temp = Tpc(ych4, yco2, yn2, yh2s);
       return(t/temp);
    }

    public double Ay(double deltax, double deltaz)              //fun17
    {
       return (deltax*deltaz);
    }

    public double Pgsc(double ych4, double yco2, double yn2, double yh2s, double t)throws
IOException, BiffException, RowsExceededException, WriteException            //fun18
    {
       t=128;
       return (((14.7)*(Pg( ych4, yco2, yn2, yh2s))*28.96/(10.72*t)));
    }

    public double Bg(double pm, double t, double ych4, double yco2, double yn2, double yh2s)
throws IOException, BiffException, RowsExceededException, WriteException          //fun19
    {
       return ((0.02827*(Z(pm, ych4, yco2, yn2, yh2s))* t/pm));
    }

    public double Phif(double phio, double p, double pf) throws IOException, BiffException,
RowsExceededException, WriteException              //fun20
    {
       return (phio*(Math.exp(Cm(p)*(p-pf))));
```

```
    }


    private static XYDataset createDataset(double pm[][][][], int t,int i, int j, int k)
    {

        DefaultXYDataset ds = new DefaultXYDataset();
        double[][] data = new double[2][t+1];

        for(int x = 0; x<=1; x++)
        {
            for(int y =0; y<=t;y++)
            {
                if(x==0)
                    data[x][y] = y;
                else
                    data[x][y]= pm[y][i][j][k];
            }
        }

        ds.addSeries("series of"+"("+(i+1)+","+(j+1)+","+(k+1)+")", data);
        return ds;
    }
    public void main() throws IOException, BiffException, RowsExceededException,
WriteException                              // Mains starts from here
    {
       System.out.print('\u000C');
        System.out.println("THE MATRIX");
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        double deltax,deltay,deltaz,deltatt,p,vl,pl; //decleration of input
        System.out.println("Enter Delta X");
        deltax = Double.parseDouble(br.readLine());
        System.out.println("Enter Delta Y");
        deltay = Double.parseDouble(br.readLine());
        System.out.println("Enter Delta Z");
        deltaz = Double.parseDouble(br.readLine());
        System.out.println("Enter total time in days");
        deltatt = Double.parseDouble(br.readLine());
        System.out.println("Enter the minimum interval of time in days");
        double ti = Double.parseDouble(br.readLine());          //decleration of time interval
        deltatt/=ti;
        deltatt = Math.round(deltatt);  // time devided by ti and rounding off the time
        int deltat = (int) deltatt;
        System.out.println("Enter the initial pressure Pm");
        p = Double.parseDouble(br.readLine());  // initial pressure input
        double t;                       //decleration of temprature
```

```
System.out.println("Enter the reservoir temprature in Farahnite");
t = Double.parseDouble(br.readLine());     //initial reservoir temp in rankine
System.out.println("Enter the value of porousity");
double phiom = Double.parseDouble(br.readLine());
System.out.println("Enter the vlaue of Vl");
vl = Double.parseDouble(br.readLine());
System.out.println("Enter the value of Pl");
pl = Double.parseDouble(br.readLine());
System.out.println("Enter the pressure of wellbore");
double wbp = Double.parseDouble(br.readLine());
double deltaywb = 1;
double deltaymat = deltay + ((deltay - deltaywb)/2) ;
double ych4,yco2,yn2,yh2s;        //input mole fractions
System.out.println("Enter the mole fraction of Methane");
ych4 = Double.parseDouble(br.readLine());
System.out.println("Enter the mole fraction of Carbon di oxide");
yco2 = Double.parseDouble(br.readLine());
System.out.println("Enter the mole fraction of Nitrogen");
yn2 = Double.parseDouble(br.readLine());
System.out.println("Enter the mole fraction of Hydrogen sulphide");
yh2s = Double.parseDouble(br.readLine());
int time = 0;
      double[][][][] pm = new double[deltat +2][9][9][9];     // the pm matrix  5X5
for(int i=0;i<9;i++)
{
   for(int j=0; j<9;j++)
   {
      for(int k=0;k<9;k++)
      {
         pm[time][i][j][k] = p;
      }

   }

}

for(int i =0;i<9;i++)
{
   pm[time][4][4][i]= wbp;
}


// take the loop for time in here.....................
double[][][] b = new double[9][9][9];  //decleration of matrix of all coeff of main eqn
double[][][] s = new double[9][9][9];  //decleration of matrix of all coeff of main eqn
double[][][] w = new double[9][9][9];  //decleration of matrix of all coeff of main eqn
```

```
double[][][] e = new double[9][9][9];  //decleration of matrix of all coeff of main eqn
double[][][] v = new double[9][9][9];  //decleration of matrix of all coeff of main eqn
double[][][] a = new double[9][9][9];  //decleration of matrix of all coeff of main eqn
double[][][] ug = new double[9][9][9];  //decleration of matrix of all coeff of main eqn
double[][][] x = new double[9][9][9];  //decleration of matrix of all coeff of main matrix eqn
double[][][] q = new double[9][9][9];  //decleration of matrix of all coeff of main matrix eqn
double[][][] c = new double[9][9][9];  //decleration of matrix of all coeff of main matrix eqn
while(time<=deltat)
{

   for(int i=0;i<9;i++)
   {
      for(int j=0; j<9;j++)
      {
         for(int k=0;k<9;k++)
         {
            double tempo = pm[time][i][j][k];
            ug[i][j][k] = Ug(tempo,t,ych4,yco2,yn2,yh2s);
         }

      }

   }

   for(int i=0;i<9;i++)
   {
      for(int j=0; j<9;j++)
      {
         for(int k=0;k<9;k++)
         {

            if(i==4 && j==4 )// && j>0 &&j<8
            {
               double tempo;
               if( i<0 || j<0 || (k-1)<0 || i>=9 || j>=9 || (k-1)>=9)
               {
                  b[i][j][k] = 0;
               }
               else
               {
                  tempo = pm[time][i][j][k-1];
                  b[i][j][k] = (((Km(tempo))*(Az(deltax,deltaywb))*(Tgx(tempo, t, ych4,
yco2, yn2, yh2s)))/deltaz);
               }
            }
            else if(i==4 &&( j==3 || j==5 ) )  //&& j>0 &&j<4
```

```
                {
                   double tempo;
                   if( i<0 || j<0 || (k-1)<0 || i>=9 || j>=9 || (k-1)>=9)
                   {
                      b[i][j][k] = 0;
                   }
                   else
                   {
                      tempo = pm[time][i][j][k-1];
                      b[i][j][k] = (((Km(tempo))*(Az(deltax,deltaymat))*(Tgx(tempo, t, ych4,
yco2, yn2, yh2s)))/deltaz);
                   }
                }
                else
                {
                   double tempo;
                   if( i<0 || j<0 || (k-1)<0 || i>=9 || j>=9 || (k-1)>=9)
                   {
                      b[i][j][k] = 0;
                   }
                   else
                   {
                      tempo = pm[time][i][j][k-1];
                      b[i][j][k] = (((Km(tempo))*(Az(deltax,deltay))*(Tgx(tempo, t, ych4, yco2,
yn2, yh2s)))/deltaz);
                   }
                }
                //System.out.print(b[i][j][k]);                                          //
delete me later
             }

          }

       }

       for(int i=0;i<9;i++)
       {
          for(int j=0; j<9;j++)
          {
             for(int k=0;k<9;k++)
             {
                if(i==4 && j==4)// && j>0 &&j<4
                {
                   double tempo;
                   if( i<0 || (j-1)<0 || k<0 || i>=9 || (j-1)>=9 || k>=9)
                   {
```

```
                    s[i][j][k] = 0;
                }
                else
                {
                    tempo = pm[time][i][j-1][k];
                    s[i][j][k] = (((Km(tempo))*(Ay(deltax,deltaz))*(Tgx(tempo, t, ych4, yco2,
yn2, yh2s)))/deltaywb);
                }
            }
            else if(i==4 &&( j==3 || j==5 )) // && j>0 &&j<4
            {
                double tempo;
                if( i<0 || (j-1)<0 || k<0 || i>=9 || (j-1)>=9 || k>=9)
                {
                    s[i][j][k] = 0;
                }
                else
                {
                    tempo = pm[time][i][j-1][k];
                    s[i][j][k] = (((Km(tempo))*(Ay(deltax,deltaz))*(Tgx(tempo, t, ych4, yco2,
yn2, yh2s)))/deltaymat);
                }
            }
            else
            {
                double tempo;
                if( i<0 || (j-1)<0 || k<0 || i>=9 || (j-1)>=9 || k>=9)
                {
                    s[i][j][k] = 0;
                }
                else
                {
                    tempo = pm[time][i][j-1][k];
                    s[i][j][k] = (((Km(tempo))*(Ay(deltax,deltaz))*(Tgx(tempo, t, ych4, yco2,
yn2, yh2s)))/deltay);
                }
            }
        }

    }

}

for(int i=0;i<9;i++)
{
    for(int j=0; j<9;j++)
```

```
        {
            for(int k=0;k<9;k++)
            {
                if(i==4 && j==4)  // && j>0 &&j<4
                {
                    double tempo;
                    if( (i-1)<0 || j<0 || k<0 || (i-1)>=9 || j>=9 || k>=9)
                    {
                        w[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pm[time][i-1][j][k];
                        w[i][j][k] = (((Km(tempo))*(Ax(deltaywb,deltaz))*(Tgx(tempo, t, ych4,
yco2, yn2, yh2s)))/deltax);
                    }
                }
                else if(i==4 && (j==3 || j==5) ) //&& j>0 &&j<4
                {
                    double tempo;
                    if( (i-1)<0 || j<0 || k<0 || (i-1)>=9 || j>=9 || k>=9)
                    {
                        w[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pm[time][i-1][j][k];
                        w[i][j][k] = (((Km(tempo))*(Ax(deltaymat,deltaz))*(Tgx(tempo, t, ych4,
yco2, yn2, yh2s)))/deltax);
                    }
                }
                else
                {
                    double tempo;
                    if( (i-1)<0 || j<0 || k<0 || (i-1)>=9 || j>=9 || k>=9)
                    {
                        w[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pm[time][i-1][j][k];
                        w[i][j][k] = (((Km(tempo))*(Ax(deltay,deltaz))*(Tgx(tempo, t, ych4, yco2,
yn2, yh2s)))/deltax);
                    }
                }
```

```
            }

        }

    }

    for(int i=0;i<9;i++)
    {
        for(int j=0; j<9;j++)
        {
            for(int k=0;k<9;k++)
            {
                if(i==4 && j==4)  // && j>0 &&j<4
                {
                    double tempo;
                    if( (i+1)<0 || j<0 || k<0 || (i+1)>=9 || j>=9 || k>=9)
                    {
                        e[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pm[time][i+1][j][k];
                        e[i][j][k] = (((Km(tempo))*(Ax(deltaywb,deltaz))*(Tgx(tempo, t, ych4,
yco2, yn2, yh2s)))/deltax);
                    }
                }
                else if(i==4 && (j==5 || j==3)) // && j>0 &&j<4
                {
                    double tempo;
                    if( (i+1)<0 || j<0 || k<0 || (i+1)>=9 || j>=9 || k>=9)
                    {
                        e[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pm[time][i+1][j][k];
                        e[i][j][k] = (((Km(tempo))*(Ax(deltaymat,deltaz))*(Tgx(tempo, t, ych4,
yco2, yn2, yh2s)))/deltax);
                    }
                }
                else
                {
                    double tempo;
                    if( (i+1)<0 || j<0 || k<0 || (i+1)>=9 || j>=9 || k>=9)
                    {
                        e[i][j][k] = 0;
```

```
                    }
                    else
                    {
                        tempo = pm[time][i+1][j][k];
                        e[i][j][k] = (((Km(tempo))*(Ax(deltay,deltaz))*(Tgx(tempo, t, ych4, yco2,
yn2, yh2s)))/deltax);
                    }
                }

            }

        }

        for(int i=0;i<9;i++)
        {
            for(int j=0; j<9;j++)
            {
                for(int k=0;k<9;k++)
                {
                    if(i==4 && j==4)  // && j>0 &&j<4
                    {
                        double tempo;
                        if( i<0 || (j+1) <0 || k<0 || i>=9 || (j+1)>=9 || k>=9)
                        {
                            v[i][j][k] = 0;
                        }
                        else
                        {
                            tempo = pm[time][i][j+1][k];
                            v[i][j][k] = (((Km(tempo))*(Ay(deltax,deltaz))*(Tgx(tempo, t, ych4, yco2,
yn2, yh2s)))/deltaywb);
                        }
                    }
                    else if(i==4 && (j==5 || j==3))  // && j>0 &&j<4
                    {
                        double tempo;
                        if( i<0 || (j+1) <0 || k<0 || i>=9 || (j+1)>=9 || k>=9)
                        {
                            v[i][j][k] = 0;
                        }
                        else
                        {
                            tempo = pm[time][i][j+1][k];
                for(int k=0;k<9;k++)
```

```
            {
               if(i==4 && j==4)  // && j>0 &&j<4
               {
                  double tempo;
                  if( i<0 || j<0 || (k+1)<0 || i>=9 || j>=9 || (k+1)>=9)
                  {
                     a[i][j][k] = 0;
                  }
                  else
                  {
                     tempo = pm[time][i][j][k+1];
                     a[i][j][k] = (((Km(tempo))*(Az(deltax,deltaywb))*(Tgx(tempo, t, ych4,
yco2, yn2, yh2s)))/deltaz);
                  }
               }
               else if(i==4 && (j==5 || j==3)) // && j>0 &&j<4
               {
                  double tempo;
                  if( i<0 || j<0 || (k+1)<0 || i>=9 || j>=9 || (k+1)>=9)
                  {
                     a[i][j][k] = 0;
                  }
                  else
                  {
                     tempo = pm[time][i][j][k+1];
                     a[i][j][k] = (((Km(tempo))*(Az(deltax,deltaymat))*(Tgx(tempo, t, ych4,
yco2, yn2, yh2s)))/deltaz);
                  }
               }
                        }

         }

         for(int i=0;i<9;i++)
         {
            for(int j=0; j<9;j++)
            {
               for(int k=0;k<9;k++)
               {
                  if(i==4 && j==4)  // && j>0 &&j<4
                  {
                     double tempo = pm[time][i][j][k];
                     x[i][j][k] = ((-1)*(Vb(deltax, deltaywb,
deltaz))/(ti*(time+1)))*((phiom*(Cm(tempo))*((Pgsc( ych4, yco2, yn2, yh2s,t))/(5.614583*
Bg(tempo, t, ych4, yco2, yn2, yh2s))))+(phiom*(Cm(tempo))*((Vd(tempo,vl,pl))-
```

```
(Va(tempo,vl,pl)))*(Pgsc( ych4, yco2, yn2, yh2s,t)))-(((2*pl*vl)/Math.pow((pl +tempo),2))*(1-
phiom)));
                    }
                    else if(i==4 && (j==5 || j==3))  // && j>0 &&j<4
                    {
                       double tempo = pm[time][i][j][k];
                       x[i][j][k] = ((-1)*(Vb(deltax, deltaymat,
deltaz))/(ti*(time+1)))*((phiom*(Cm(tempo))*((Pgsc( ych4, yco2, yn2, yh2s,t))/(5.614583*
Bg(tempo, t, ych4, yco2, yn2, yh2s))))+(phiom*(Cm(tempo))*((Vd(tempo,vl,pl))-
(Va(tempo,vl,pl)))*(Pgsc( ych4, yco2, yn2, yh2s,t)))-(((2*pl*vl)/Math.pow((pl +tempo),2))*(1-
phiom)));
                    }
                    else
      }

      System.out.println("the pressure's are");
      for(time =0;time<=deltat;time++)
      {
         System.out.println("pressure's at time interval   "+time);
         for(int i=0; i<9;i++)
         {
            for(int j=0;j<9;j++)
            {
               for(int k=0;k<9;k++)
               {
                  System.out.println("("+(i+1)+","+(j+1)+","+(k+1)+")"+"="+pm[time][i][j][k]);


               }
            }
         }
      }

      for(int i=0; i<9;i++)
      {
         for(int j=0;j<9;j++)
         {
            for(int k=0;k<9;k++)
            {
               WritableWorkbook workbook = Workbook.createWorkbook(new
File("E:\\matrix\\Matrix"+(i+1)+""+(j+1)+""+(k+1)+".xls"));
               WritableSheet sheet = workbook.createSheet("First Sheet", 0);
               for(time =0;time<=deltat;time++)
               {

                  int tempo = time;
```

```
                Number number1 = new Number(0, tempo, time);
                sheet.addCell(number1);

                Number number = new Number(1, tempo, pm[time][i][j][k]);
                sheet.addCell(number);

            }
            workbook.write();
            workbook.close();

        }
    }
}

/*for(int i=0; i<9;i++)
{
    for(int j=0;j<9;j++)
    {
        for(int k=0;k<9;k++)
        {
            WritableWorkbook workbook = Workbook.createWorkbook(new
File("E:\\matrix\\Bg"+(i+1)+""+(j+1)+""+(k+1)+".xls"));
            WritableSheet sheet = workbook.createSheet("First Sheet", 0);
            for(time =0;time<=deltat;time++)
            {

                int tempo = time;
                double tempo2 = pm[time][i][j][k];

                Number number1 = new Number(0, tempo, time);
                sheet.addCell(number1);

                Number number = new Number(1, tempo, Bg(tempo2, t, ych4, yco2, yn2, yh2s));
                sheet.addCell(number);

            }
            workbook.write();
            workbook.close();
        }
    }
}
for(int i=0; i<9;i++)
{
    for(int j=0;j<9;j++)
    {
```

```
    for(int k=0;k<9;k++)
    {
 WritableSheet sheet = workbook.createSheet("First Sheet", 0);
        for(time =0;time<=deltat;time++)
        {
           int tempo = time;
           double tempo2 = pm[time][i][j][k];
           Number number1 = new Number(0, tempo, time);
           sheet.addCell(number1);
           Number number = new Number(1, tempo, Ug(tempo2, t, ych4, yco2, yn2, yh2s));
           sheet.addCell(number);
        }
        workbook.write();
        workbook.close();
    }
}
```

ANNEXURE-II

Compilation of nonlinear partial differential equation for the gas flow in the hydraulic or induced fractures using JAVA.

```
import java.io.*;
import java.lang.Math;
import java.util.Scanner;
import java.util.HashSet;
import javax.swing.JFrame;
import javax.swing.SwingUtilities;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.data.xy.DefaultXYDataset;
import org.jfree.data.xy.XYDataset;
import java.io.File;
import java.io.IOException;
import jxl.*;
import jxl.read.biff.BiffException;
import jxl.write.*;
import jxl.write.Number;
import jxl.write.biff.RowsExceededException;
class Fractures
{
   public static double[][] invert(double a[][])
   {
      int n = a.length;
      double x[][] = new double[n][n];
      double b[][] = new double[n][n];
      int index[] = new int[n];
      for (int i=0; i<n; ++i)
         b[i][i] = 1;
      // Transform the matrix into an upper triangle
      gaussian(a, index);
      // Update the matrix b[i][j] with the ratios stored
      for (int i=0; i<n-1; ++i)
         for (int j=i+1; j<n; ++j)
            for (int k=0; k<n; ++k)
               b[index[j]][k]
               -= a[index[j]][i]*b[index[i]][k];
```

```
   // Perform backward substitutions
   for (int i=0; i<n; ++i)
   {
      x[n-1][i] = b[index[n-1]][i]/a[index[n-1]][n-1];
      for (int j=n-2; j>=0; --j)
      {
         x[j][i] = b[index[j]][i];
         for (int k=j+1; k<n; ++k)
         {
            x[j][i] -= a[index[j]][k]*x[k][i];
         }
         x[j][i] /= a[index[j]][j];
      }
   }
   return x;
}
// Method to carry out the partial-pivoting Gaussian
// elimination.  Here index[] stores pivoting order.
public static void gaussian(double a[][], int index[])
{
   int n = index.length;
   double c[] = new double[n];
   // Initialize the index
   for (int i=0; i<n; ++i)
      index[i] = i;
   // Find the rescaling factors, one from each row
   for (int i=0; i<n; ++i)
   {
      double c1 = 0;
      for (int j=0; j<n; ++j)
      {
         double c0 = Math.abs(a[i][j]);
         if (c0 > c1) c1 = c0;
      }
      c[i] = c1;
   }

   // Search the pivoting element from each column
   int k = 0;
   for (int j=0; j<n-1; ++j)
   {
```

```
        double pi1 = 0;
        for (int i=j; i<n; ++i)
        {
           double pi0 = Math.abs(a[index[i]][j]);
           pi0 /= c[index[i]];
           if (pi0 > pi1)
           {
              pi1 = pi0;
              k = i;
           }
        }
  {
     DefaultXYDataset ds = new DefaultXYDataset();
     double[][] data = new double[2][t+1];
     for(int x = 0; x<=1; x++)
     {
        for(int y =0; y<=t;y++)
        {
           if(x==0)
              data[x][y] = y;
           else
              data[x][y]= pm[y][i][j][k];
        }
     }
     ds.addSeries("series of"+"("+(i+1)+","+(j+1)+","+(k+1)+")", data);
     return ds;
  }
  public void main() throws IOException, BiffException, RowsExceededException, WriteException
// Mains starts from here
  {
     System.out.print('\u000C');
     BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
     double deltax,deltay,deltaz,deltatt,p,vl,pl; //decleration of input
     double deltaxf,deltayf,deltattf,swf,sgm,kf,phiof,wfc,kfg;; //decleration of input
     System.out.println("Enter Delta X");
     deltax = Double.parseDouble(br.readLine());
     System.out.println("Enter Delta Y");
     deltay = Double.parseDouble(br.readLine());
     System.out.println("Enter Delta Z for matrix");
     deltaz = Double.parseDouble(br.readLine());
     System.out.println("The value of Delta Z for fracture is 1");
```

```
double deltazf = 1;
System.out.println("Enter total time in days");
deltatt = Double.parseDouble(br.readLine());
System.out.println("Enter the minimum interval of time in days");
double ti = Double.parseDouble(br.readLine());          //decleration of time interval
deltatt/=ti;
deltatt = Math.round(deltatt);  // time devided by ti and rounding off the time
int deltat = (int) deltatt;
System.out.println("Enter the initial pressure Pm");
p = Double.parseDouble(br.readLine());  // initial pressure input
double t;                       //decleration of temprature
System.out.println("Enter the reservoir temprature in Farahnite");
t = Double.parseDouble(br.readLine());    //initial reservoir temp in rankine
System.out.println("Enter the value of porousity of the matrix");
double phiom = Double.parseDouble(br.readLine());
System.out.println("Enter the vlaue of Vl");
vl = Double.parseDouble(br.readLine());
System.out.println("Enter the value of Pl");
pl = Double.parseDouble(br.readLine());
  xf[i][j][k] = (((2/wfc)*kfg*(Tgx(tempo, tf, ych4f, yco2f, yn2f, yh2sf))*deltaxf*deltaymat) + (((Vb(deltaxf,
deltaymat, deltazf) * Pgsc(ych4f, yco2f, yn2f, yh2sf,tf))/(5.614583* Bg(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))*(1-swf)*(phiof)*Cm(tempo)/(tif*(timef+2))
   }
   else
      {
                        double tempo = pmf[timef][i][j][k];
                        xf[i][j][k] = (((2/wfc)*kfg*(Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf))*deltaxf*deltayf) + (((Vb(deltaxf, deltayf, deltazf) * Pgsc(ych4f, yco2f, yn2f, yh2sf,tf))/(5.614583*
Bg(tempo, tf, ych4f, yco2f, yn2f, yh2sf)))*(1-swf)*(phiof)*Cm(tempo)/(tif*(timef+2))));
                     }
                  }
                  else  // its not a fracture
                  {
                     if(i==4 && j==4)  // && j>0 &&j<4
                     {
                        double tempo = pmf[timef][i][j][k];
                        xf[i][j][k] = ((-1)*(Vb(deltaxf, deltaywb,
deltaz))/(tif*(timef+2)))*((phiom*(Cm(tempo))*((Pgsc( ych4f, yco2f, yn2f, yh2sf,tf))/(5.614583*
Bg(tempo, tf, ych4f, yco2f, yn2f, yh2sf))))+(phiom*(Cm(tempo))*((Vd(tempo,vl,pl))-
(Va(tempo,vl,pl)))*(Pgsc( ych4f, yco2f, yn2f, yh2sf,tf)))-(((2*pl*vl)/Math.pow((pl +tempo),2))*(1-
phiom)));
```

```
                    }
                    else if(i==4 && (j==5 || j==3))  // && j>0 &&j<4
                    {
                       double tempo = pmf[timef][i][j][k];
                       xf[i][j][k] = ((-1)*(Vb(deltaxf, deltaymat,
deltaz))/(tif*(timef+2)))*((phiom*(Cm(tempo))*((Pgsc( ych4f, yco2f, yn2f, yh2sf,tf))/(5.614583*
Bg(tempo, tf, ych4f, yco2f, yn2f, yh2sf))))+(phiom*(Cm(tempo))*((Vd(tempo,vl,pl))-
(Va(tempo,vl,pl)))*(Pgsc( ych4f, yco2f, yn2f, yh2sf,tf)))-(((2*pl*vl)/Math.pow((pl +tempo),2))*(1-
phiom)));
                    }
                    else
                    {
                       double tempo = pmf[timef][i][j][k];
                       xf[i][j][k] = ((-1)*(Vb(deltaxf, deltayf,
deltaz))/(tif*(timef+2)))*((phiom*(Cm(tempo))*((Pgsc( ych4f, yco2f, yn2f, yh2sf,tf))/(5.614583*
Bg(tempo, tf, ych4f, yco2f, yn2f, yh2sf))))+(phiom*(Cm(tempo))*((Vd(tempo,vl,pl))-
(Va(tempo,vl,pl)))*(Pgsc( ych4f, yco2f, yn2f, yh2sf,tf)))-(((2*pl*vl)/Math.pow((pl +tempo),2))*(1-
phiom)));

                    }
                 }
              }

           }

        }

     case 5:
     {
        for(int i=0;i<9;i++)
        {
           for(int j=0; j<9;j++)
           {
              for(int k=0;k<9;k++)
              {

                 if( i==4 && ( k==0 || k==2 || k==4 || k==6 || k==8))  // if its a fracture
                 {
                    bf[i][j][k] = 0;
                 }
                 else // it is not a fracture
```

```
                {
                    if(i==4 && j==4 )
                    {
                        double tempo;
                        if( i<0 || j<0 || (k-1)<0 || i>=9 || j>=9 || (k-1)>=9)
                        {
                            bf[i][j][k] = 0;
                        }
                        else
                        {
                            tempo = pmf[timef][i][j][k-1];
                            bf[i][j][k] = (((Km(tempo))*(Az(deltaxf,deltaywb))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaz);
                        }
                    }
                    else if(i==4 &&( j==3 || j==5 ) )  //&& j>0 &&j<4
                    {
                        double tempo;
                        if( i<0 || j<0 || (k-1)<0 || i>=9 || j>=9 || (k-1)>=9)
                        {
                            bf[i][j][k] = 0;
                        }
                        else
                        {
                            tempo = pmf[timef][i][j][k-1];
                            bf[i][j][k] = (((Km(tempo))*(Az(deltaxf,deltaymat))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaz);
                        }
                    }
                    else
                    {
                        double tempo;
                        if( i<0 || j<0 || (k-1)<0 || i>=9 || j>=9 || (k-1)>=9)
                        {
                            bf[i][j][k] = 0;
                        }
                        else
                        {
                            tempo = pm[timef][i][j][k-1];
                            bf[i][j][k] = (((Km(tempo))*(Az(deltaxf,deltayf))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaz);
```

```
                    }
                  }
                }

              }
            }
          }

          for(int i=0;i<9;i++)
          {
            for(int j=0; j<9;j++)
            {
              for(int k=0;k<9;k++)
              {
                if ( i==4 && ( k==0 || k==2 || k==4 || k==6 || k==8))  // its a fracture
                {
                  if(i==4 && j==4)// && j>0 &&j<4
                  {
                    double tempo;
                    if( i<0 || (j-1)<0 || k<0 || i>=9 || (j-1)>=9 || k>=9)
                    {
                      sf[i][j][k] = 0;
                    }
                    else
                    {
                      tempo = pmf[timef][i][j-1][k];
                      sf[i][j][k] = ((kf*(Ay(deltaxf,deltazf))*(kfg*Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))/deltaywb);
                    }
                  }
                  else if(i==4 &&( j==3 || j==5 )) // && j>0 &&j<4
                  {
                    double tempo;
                    if( i<0 || (j-1)<0 || k<0 || i>=9 || (j-1)>=9 || k>=9)
                    {
                      sf[i][j][k] = 0;
                    }
                    else
                    {
                      tempo = pmf[timef][i][j-1][k];
```

```
                        sf[i][j][k] = ((kf*(Ay(deltaxf,deltazf))*(kfg*Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))/deltaymat);
                    }
                }
                else
                {
                    double tempo;
                    if( i<0 || (j-1)<0 || k<0 || i>=9 || (j-1)>=9 || k>=9)
                    {
                        sf[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i][j-1][k];
                        sf[i][j][k] = ((kf*(Ay(deltaxf,deltazf))*(kfg*Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))/deltayf);
                    }
                }
            }

            else  // its not a fracture
            {
                if(i==4 && j==4)// && j>0 &&j<4
                {
                    double tempo;
                    if( i<0 || (j-1)<0 || k<0 || i>=9 || (j-1)>=9 || k>=9)
                    {
                        sf[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i][j-1][k];
                        sf[i][j][k] = (((Km(tempo))*(Ay(deltaxf,deltaz))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaywb);
                    }
                }
                else if(i==4 &&( j==3 || j==5 )) // && j>0 &&j<4
                {
                    double tempo;
                    if( i<0 || (j-1)<0 || k<0 || i>=9 || (j-1)>=9 || k>=9)
                    {
```

```
                        sf[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i][j-1][k];
                        sf[i][j][k] = (((Km(tempo))*(Ay(deltaxf,deltaz))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaymat);
                    }
                }
                else
                {
                    double tempo;
                    if( i<0 || (j-1)<0 || k<0 || i>=9 || (j-1)>=9 || k>=9)
                    {
                        sf[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i][j-1][k];
                        sf[i][j][k] = (((Km(tempo))*(Ay(deltaxf,deltaz))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltayf);
                    }
                }
            }
        }
    }

    }

    for(int i=0;i<9;i++)
    {
        for(int j=0; j<9;j++)
        {
            for(int k=0;k<9;k++)
            {
                if ( i==4 && ( k==0 || k==2 || k==4 || k==6 || k==8))  // its a fracture
                {
                    if(i==4 && j==4)
                    {
                        double tempo;
                        if( (i-1)<0 || j<0 || k<0 || (i-1)>=9 || j>=9 || k>=9)
```

```
                    {
                        wf[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i-1][j][k];
                        wf[i][j][k] = ((kf*(Ax(deltaywb,deltazf))*(kfg*Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))/deltaxf);
                    }
                }
                else if(i==4 && (j==3 || j==5) ) //&& j>0 &&j<4
                {
                    double tempo;
                    if( (i-1)<0 || j<0 || k<0 || (i-1)>=9 || j>=9 || k>=9)
                    {
                        wf[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i-1][j][k];
                        wf[i][j][k] = ((kf*(Ax(deltaymat,deltazf))*(kfg*Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))/deltaxf);
                    }
                }
                else
                {
                    double tempo;
                    if( (i-1)<0 || j<0 || k<0 || (i-1)>=9 || j>=9 || k>=9)
                    {
                        wf[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i-1][j][k];
                        wf[i][j][k] = ((kf*(Ax(deltayf,deltazf))*(kfg*Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))/deltaxf);
                    }
                }
            }
            else  // its not a fracture
            {
```

```
                    if(i==4 && j==4)
                    {
                       double tempo;
                       if( (i-1)<0 || j<0 || k<0 || (i-1)>=9 || j>=9 || k>=9)
                       {
                          wf[i][j][k] = 0;
                       }
                       else
                       {
                          tempo = pmf[timef][i-1][j][k];
                          wf[i][j][k] = (((Km(tempo))*(Ax(deltaywb,deltaz))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaxf);
                       }
                    }
                    else if(i==4 && (j==3 || j==5) ) //&& j>0 &&j<4
                    {
                       double tempo;
                       if( (i-1)<0 || j<0 || k<0 || (i-1)>=9 || j>=9 || k>=9)
                       {
                          wf[i][j][k] = 0;
                       }
                       else
                       {
                          tempo = pmf[timef][i-1][j][k];
                          wf[i][j][k] = (((Km(tempo))*(Ax(deltaymat,deltaz))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaxf);
                       }
                    }
                    else
                    {
                       double tempo;
                       if( (i-1)<0 || j<0 || k<0 || (i-1)>=9 || j>=9 || k>=9)
                       {
                          wf[i][j][k] = 0;
                       }
                       else
                       {
                          tempo = pmf[timef][i-1][j][k];
                          wf[i][j][k] = (((Km(tempo))*(Ax(deltayf,deltaz))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaxf);
                       }
```

```
                }
            }
        }

    }

}

for(int i=0;i<9;i++)
{
    for(int j=0; j<9;j++)
    {
        for(int k=0;k<9;k++)
        {
            if ( i==4 && ( k==0 || k==2 || k==4 || k==6 || k==8))  // its a fracture
            {
                if(i==4 && j==4)
                {
                    double tempo;
                    if( (i+1)<0 || j<0 || k<0 || (i+1)>=9 || j>=9 || k>=9)
                    {
                        ef[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i+1][j][k];
                        ef[i][j][k] = ((kf*(Ax(deltaywb,deltazf))*(kfg*Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))/deltaxf);
                    }
                }
                else if(i==4 && (j==5 || j==3)) // && j>0 &&j<4
                {
                    double tempo;
                    if( (i+1)<0 || j<0 || k<0 || (i+1)>=9 || j>=9 || k>=9)
                    {
                        ef[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i+1][j][k];
```

```
                        ef[i][j][k] = ((kf*(Ax(deltaymat,deltazf))*(kfg*Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))/deltaxf);
                    }
                }
                else
                {
                    double tempo;
                    if( (i+1)<0 || j<0 || k<0 || (i+1)>=9 || j>=9 || k>=9)
                    {
                        ef[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i+1][j][k];
                        ef[i][j][k] = ((kf*(Ax(deltayf,deltazf))*(kfg*Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))/deltaxf);
                    }
                }
            }
            else  // its not a fracture
            {
                if(i==4 && j==4)
                {
                    double tempo;
                    if( (i+1)<0 || j<0 || k<0 || (i+1)>=9 || j>=9 || k>=9)
                    {
                        ef[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i+1][j][k];
                        ef[i][j][k] = (((Km(tempo))*(Ax(deltaywb,deltaz))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaxf);
                    }
                }
                else if(i==4 && (j==5 || j==3)) // && j>0 &&j<4
                {
                    double tempo;
                    if( (i+1)<0 || j<0 || k<0 || (i+1)>=9 || j>=9 || k>=9)
                    {
                        ef[i][j][k] = 0;
```

```
                        }
                        else
                        {
                            tempo = pmf[timef][i+1][j][k];
                            ef[i][j][k] = (((Km(tempo))*(Ax(deltaymat,deltaz))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaxf);
                        }
                    }
                    else
                    {
                        double tempo;
                        if( (i+1)<0 || j<0 || k<0 || (i+1)>=9 || j>=9 || k>=9)
                        {
                            ef[i][j][k] = 0;
                        }
                        else
                        {
                            tempo = pmf[timef][i+1][j][k];
                            ef[i][j][k] = (((Km(tempo))*(Ax(deltayf,deltaz))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaxf);
                        }
                    }
                }
            }

        }

    }

    for(int i=0;i<9;i++)
    {
        for(int j=0; j<9;j++)
        {
            for(int k=0;k<9;k++)
            {
                if ( i==4 && ( k==0 || k==2 || k==4 || k==6 || k==8))  // its a fracture
                {
                    if(i==4 && j==4)  // && j>0 &&j<4
                    {
                        double tempo;
                        if( i<0 || (j+1) <0 || k<0 || i>=9 || (j+1)>=9 || k>=9)
```

```
                    {
                        vf[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i][j+1][k];
                        vf[i][j][k] = ((kf*(Ay(deltaxf,deltazf))*(kfg*Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))/deltaywb);
                    }
                }
                else if(i==4 && (j==5 || j==3))  // && j>0 &&j<4
                {
                    double tempo;
                    if( i<0 || (j+1) <0 || k<0 || i>=9 || (j+1)>=9 || k>=9)
                    {
                        vf[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i][j+1][k];
                        vf[i][j][k] = ((kf*(Ay(deltaxf,deltazf))*(kfg*Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))/deltaymat);
                    }
                }
                else
                {
                    double tempo;
                    if( i<0 || (j+1) <0 || k<0 || i>=9 || (j+1)>=9 || k>=9)
                    {
                        vf[i][j][k] = 0;
                    }
                    else
                    {
                        tempo = pmf[timef][i][j+1][k];
                        vf[i][j][k] = ((kf*(Ay(deltaxf,deltazf))*(kfg*Tgx(tempo, tf, ych4f, yco2f, yn2f,
yh2sf)))/deltayf);
                    }
                }
            }
            else  // its not a fracture
            {
```

```
                    if(i==4 && j==4)  // && j>0 &&j<4
                    {
                       double tempo;
                       if( i<0 || (j+1) <0 || k<0 || i>=9 || (j+1)>=9 || k>=9)
                       {
                          vf[i][j][k] = 0;
                       }
                       else
                       {
                          tempo = pmf[timef][i][j+1][k];
                          vf[i][j][k] = (((Km(tempo))*(Ay(deltaxf,deltaz))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaywb);
                       }
                    }
                    else if(i==4 && (j==5 || j==3))  // && j>0 &&j<4
                    {
                       double tempo;
                       if( i<0 || (j+1) <0 || k<0 || i>=9 || (j+1)>=9 || k>=9)
                       {
                          vf[i][j][k] = 0;
                       }
                       else
                       {
                          tempo = pmf[timef][i][j+1][k];
                          vf[i][j][k] = (((Km(tempo))*(Ay(deltaxf,deltaz))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltaymat);
                       }
                    }
                    else
                    {
                       double tempo;
                       if( i<0 || (j+1) <0 || k<0 || i>=9 || (j+1)>=9 || k>=9)
                       {
                          vf[i][j][k] = 0;
                       }
                       else
                       {
                          tempo = pmf[timef][i][j+1][k];
                          vf[i][j][k] = (((Km(tempo))*(Ay(deltaxf,deltaz))*(Tgx(tempo, tf, ych4f, yco2f,
yn2f, yh2sf)))/deltayf);
                       }
```

```
            }
          }
        }
      }

}

for(int i=0;i<9;i++)
{
   for(int j=0; j<9;j++)
   {
      for(int k=0;k<9;k++)
      {
         if ( i==4 && ( k==0 || k==2 || k==4 || k==6 || k==8))  // its a fracture
         {

            af[i][j][k] = 0;

         }
         else  // its not a fracture
         {
            if(i==4 && j==4)  // && j>0 &&j<4
            {
               double tempo;
               if( i<0 || j<0 || (k+1)<0 || i>=9 || j>=9 || (k+1)>=9)
               {
                  af[i][j][k] = 0;
               }
               else
```