```
/*****************************************************
 * File: Prediction.java
 * created 05-Apr-01 3:45:47 PM by Jina
 */

import java.util.Random;

public class Prediction extends Thread
{
      MyLp  Solver;
      int prog, inkr=0;
      boolean Stop = false, red= false;
      double[] TAll =new double[ARABS.lenght];
      double[] PAll =new double[ARABS.lenght];
      int M=ARABS.class0;
      //-----------------------------------------------------------Predict
    public double Predict()
      {
            double ww;
            double temp;
            ww=0;
            for ( int i=1;i<=ARABS.class1;i++)
            {
                  //System.out.println("ARABS.class2 "+ARABS.class2);
                  if (i> ARABS.class2)
                  {
                        temp=0;
                        ww= ww;
                   }
                  else
                  {
                        //System.out.println("i----- ciklas "+i);
                        for ( int m=1;m<=M;m++)
                        {
                              if(m==1)
                                    temp= ARABS.past[M*( ARABS.class2-i+1)-(M-m)];
                              else
                                    temp= ARABS.past[M*( ARABS.class2-i+2)-(M-m)];
                              ww=ww + (ARABS.Acoef[M*i-(M-m)])*temp;
                              //System.out.println("temp*a [] "+temp+ " * "+
(ARABS.Acoef[M*i-(M-m)]));
                              //System.out.println("ww "+ww);
                        }
                        //ww=Math.abs(ww);
                  }
             }
            return ww;
      }
      //-------------------------------------------------------------------
PredictMulti
      public double[] PredictMulti()
      {
            double[] pred =new double[4000];
            int tempT=ARABS.class2;
       // System.out.println("ARABS.class2 "+ARABS.class2);
            for (int i=tempT; i<=(tempT+ARABS.pgclass6); i++)
            {
                   ARABS.class2=i;
                   {
```

```
            //          pred[i-tempT+1]=Math.abs( Predict());
                        pred[i-tempT+1]=Predict();
                  //System.out.println("ww "+(i-tempT+1)+"    "+pred[i-tempT+1]);
                    }
            }
            ARABS.class2=tempT;
            //System.out.println("pred "+pred);
            return pred;
      }
      //-----------------------------------------------------------------
Residuals
      public double Residuals()
      {
            double sigma=0.0,average=0.0;
            M=ARABS.class0;
            ARABS.Residuals[1]= ARABS.past[1];
            sigma=0.0;
            for ( int i=2;i<= ARABS.class2;i++)
            {
                  ARABS.Residuals[i]=ARABS.past[(i-1)*M+1];
                  for ( int j=1;j<=ARABS.class1;j++)
                  {
                        if ((i-j)>0)
                        {
                              for ( int m=1;m<=M;m++)
                                    ARABS.Residuals[i]=ARABS.Residuals[i]-
ARABS.Acoef[M*j-(M-m)]*(ARABS.past[M*( i-j)-(M-m)]);
                        }
                  }
                  average = average +(ARABS.Residuals[i]);
            }
            average = average/(ARABS.class2-1);
             for ( int i=2;i<= ARABS.class2;i++)
            {
                  sigma=sigma+(ARABS.Residuals[i]-average)*(ARABS.Residuals[i]-
average);
            }
            sigma/=(ARABS.class2-1);
          sigma = Math.sqrt( sigma );
            return sigma;
      }
      public double Residuals1()
      {
            double sigma=0.0,average=0.0;
            M=ARABS.class0;
            sigma=0.0;
            for ( int i=1;i<= ARABS.class2;i++)
            {
                  average = average +ARABS.past[(i-1)*M+1];

            }
            average = average/(ARABS.class2-1);
             for ( int i=1;i<= ARABS.class2;i++)
            {
                  sigma=sigma+(ARABS.past[(i-1)*M+1]-average)*(ARABS.past[(i-1)*M
+1]-average);
            }
            sigma/=(ARABS.class2-2);
          sigma = Math.sqrt( sigma );
```

```
            return sigma;
      }
      //-----------------------------------------------------------------
      static int iset = 0;
      static double gset;
      static Random generator = new Random();

      public static void setSeed( int seed ) {
            generator.setSeed( seed );
      }
    //-----------------------------------------------------------------
    public static double RAND() {
      double r;
                r=Math.random();
      return r;//generator.nextDouble();
  }
      //-----------------------------------------------------------------
      public static double gauss() {
        double fac, r, v1, v2;

        if (iset == 0) {
            do {
              v1 = 2.0 * RAND() - 1.0;
              v2 = 2.0 * RAND() - 1.0;
              r = v1 * v1 + v2 * v2;
            } while (r >= 1.0);

            fac = Math.sqrt(-2.0 * Math.log(r) / r);
            gset = v1 * fac;
            iset = 1;
            return v2 * fac;
        }
        else {
            iset = 0;
            return gset;
        }
      }
       //-----------------------------------------------------------------------
       public double Noise() {

          double noise=0;

              long ltime = System.currentTimeMillis() / 1000; // = time(null);
              int stime = (int) ltime / 2;

              setSeed( stime );       //srand(stime);
              noise = gauss();

    return noise;
      }
}
```