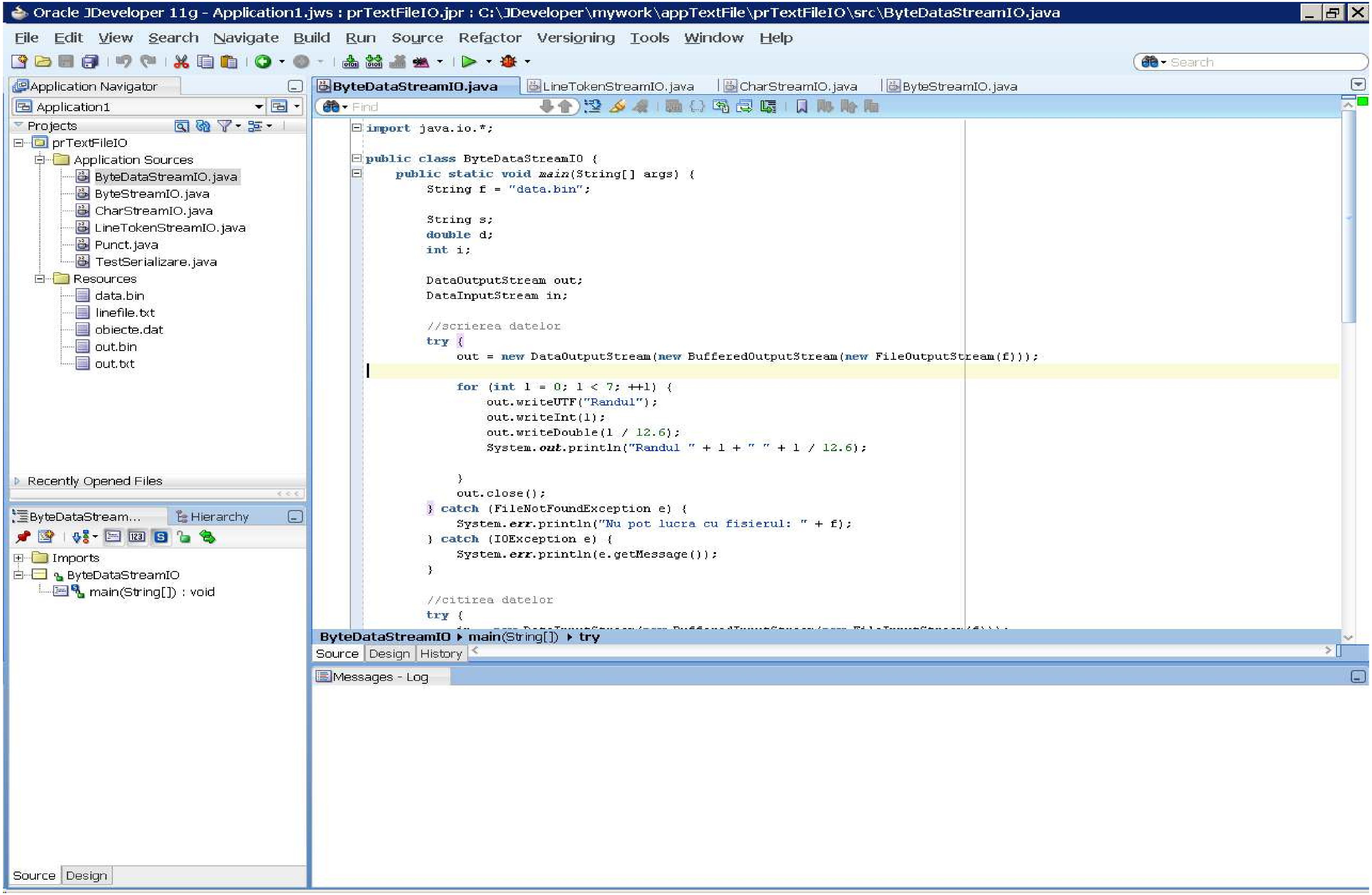

```
import java.io.*;

public class ByteDataStreamIO {
    public static void main(String[] args) {
        String f = "data.bin";

        String s;
        double d;
        int i;

        DataOutputStream out;
        DataInputStream in;

        //scrierea datelor
        try {
            out = new DataOutputStream(new BufferedOutputStream(new FileOutputStream(f)));

            for (int l = 0; l < 7; ++l) {
                out.writeUTF("Randul");
                out.writeInt(l);
                out.writeDouble(l / 12.6);
                System.out.println("Randul " + l + " " + l / 12.6);

            }
            out.close();
        } catch (FileNotFoundException e) {
            System.err.println("Nu pot lucra cu fisierul: " + f);
        } catch (IOException e) {
            System.err.println(e.getMessage());
        }

        //citirea datelor
        try {
```

AppFileIO > PrFileIO> ByteDataStreamIO

```
import java.io.*;

public class ByteDataStreamIO {
    public static void main(String[] args) {
        String f = "data.bin";

        String s;
        double d;
        int i;

        DataOutputStream out;
        DataInputStream in;

        //scrierea datelor
        try {
            out = new DataOutputStream(new BufferedOutputStream(new
FileOutputStream(f)));

            for (int l = 0; l < 7; ++l) {
                out.writeUTF("Randul");
                out.writeInt(l);
                out.writeDouble(l / 12.6);
                System.out.println("Randul " + l + " " + l / 12.6);

            }
            out.close();
        } catch (FileNotFoundException e) {
            System.err.println("Nu pot lucra cu fisierul: " + f);
        } catch (IOException e) {
            System.err.println(e.getMessage());
        }

        //citirea datelor
        try {
            in = new DataInputStream(new BufferedInputStream(new
FileInputStream(f)));

            for (int l = 0; l < 7; ++l) {
                s = in.readUTF();
                i = in.readInt();
                d = in.readDouble();
                System.out.println("S-a citit: " + s + " " + i + " " + d);
            }
            in.close();
        } catch (FileNotFoundException e) {
            System.err.println("Nu pot lucra cu fisierul: " + f);
        } catch (IOException e) {
            System.err.println(e.getMessage());
        }
    }

}
```

AppFileIO > PrFileIO> ByteStreamIO

```
import java.io.*;

public class ByteStreamIO {

    public static void main(String[] args) {

        int nroct = 0;
        int iout = 2807;
        byte[] bout = new byte[2];

        try {
            FileOutputStream scriebin = new FileOutputStream("out.bin");

            //transformat int-ul in byte-uri
            bout[0] = (byte)(iout / 256); //10
            bout[1] = (byte)(iout % 256); //247

            scriebin.write(bout);
            scriebin.close();

            FileInputStream citestebin = new FileInputStream("out.bin");
            nroct = citestebin.available();
            System.out.println(nroct + " octeti.");
            if (nroct > 0) {
                /* se poate si asa daca
                 * nu dorim ciclu
                 byte[] bin = new byte[nroct];
                 citestebin.read(bin);
                */

                for (int i = 0; i < nroct; i++)
                    System.out.println(citestebin.read());

                /* se poate si asa desi numarul
                 * de octeti se cunoaste

                while ((bin = (byte)citestebin.read()) != -1)
                    System.out.println((int) bin);
                */
            }
            citestebin.close();

        } catch (FileNotFoundException n) {
            System.err.println("Nu pot lucra cu fiserul!");
        } catch (IOException e) {
            System.err.println("Eroarea de I/E: " + e);
        }
    }
```

AppFileIO > PrFileIO> CharStreamIO

```
import java.io.*;

public class CharStreamIO {

    public static void main(String[] args) {
        String s = "2807";
        int nroct = 0, in;


        try {
            FileWriter scrietxt = new FileWriter("out.txt");

            for(int i=0; i<s.length();++i)
                scrietxt.write(s.charAt(i));
            /* se poate scrie direct tot sirul cu:
              scrietxt.write(s);

            */
            scrietxt.flush(); // inainte de close nu e necesar
            scrietxt.close();

            FileReader citestetxt = new FileReader("out.txt");

            while ((in = citestetxt.read()) != -1) {
                System.out.println((char)in);
                nroct++;
            }

            citestetxt.close();

        } catch (IOException e) {
            System.out.println("Eroarea de I/E: " + e);
        }

    }
}
```

AppFileIO > PrFileIO> LineTokenStreamIO

```
import java.io.*;
import java.util.StringTokenizer;

public class LineTokenStreamIO {
    public static void main(String[] args) {
        int l = 0;
        String linie = null;
        PrintWriter fout = null;
        BufferedReader fin = null;
        String s;
        int x;
        double y1, y2;

        try {
            fout = new PrintWriter(new FileWriter("linefile.txt"));

            //scrierea datelor in fisier, separatorul este ","
            for (int i = 0; i < 5; ++i)
                fout.println("Linia " + (i + 1) + "," + i + "," + Math.sin(i) +
                             "," + Math.cos(i));
            fout.close();

            fin = new BufferedReader(new FileReader("linefile.txt"));
            while ((linie = fin.readLine()) != null) {
                System.out.println("S-a citit linia: "+linie);

                //extragerea simbolurilor pe baza separatorului ,
                StringTokenizer t = new StringTokenizer(linie,",");
                s = t.nextToken();
                x = Integer.parseInt(t.nextToken());
                y1 = Double.parseDouble(t.nextToken());
                y2 = Double.parseDouble(t.nextToken());
                System.out.println(s+x+y1+y2);
                ++l;
            }
            System.out.println("S-au citit si prelucrat" + l + " linii");

            fin.close();

        } catch (IOException e) {
        }
    }
}
```

AppFileIO > PrFileIO> Punct

```
import java.io.Serializable;

public class Punct implements Serializable{
//Campuri
   private double x;
   private double y;
   private double distanta;

//Constructori
   Punct() {
      setX(0);
      setY(0);
      distanta = 0;
   }

   Punct(double x, double y) {
      setX(x);
      setY(y);
      actualizareDistanta();
   }

// Metode
   public void setX(double x) {
      this.x = x;
      actualizareDistanta();
   }

   public void setY(double y) {
      this.y = y;
      actualizareDistanta();
   }

   public double x() {
      return x;
   }

   public double y() {
         return y;
   }

   public double distantaOrigine() {
      return distanta;
   }

   private void actualizareDistanta() {
      distanta = Math.sqrt(x*x+y*y);
   }

    public String toString() {
      return "<" + x + "," + y + ">";
   }
}
```

AppFileIO > PrFileIO> TestSerializare

```
import java.io.*;

public class TestSerializare {
   public static void main(String[] args) {
      Punct p1 = new Punct(1,1);

      Punct[] puncte = new Punct[3];

      puncte[0]=p1;
      puncte[1]= new Punct(2,2);
      puncte[2]= new Punct(3,3);

      ObjectOutputStream out;
```

```
        ObjectInputStream in;


        try {
            out = new ObjectOutputStream(new
FileOutputStream("obiecte.dat"));
            out.writeObject(puncte);
            out.close();

            in = new ObjectInputStream(new FileInputStream("obiecte.dat"));
            Punct [] punctecitie = (Punct []) in.readObject();
            in.close();

            for (Punct p: punctecitie)
                System.out.println(p);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```