Lösningsförslag till övning 5.

Uppgift 1

a)

```
import java.io.*;
public class Faulty {
    public static void main(String[] arg) throws IOException {
        BufferedReader infil = new BufferedReader(new FileReader("data.txt"));
        int totally = 0;
        int faultyBefore = 0;
        int faultyAfter = 0;
        while(true) {
            String rad = infil.readLine();
            if (rad == null)
                break;
            totally++;
            double diff = Double.parseDouble(rad);
            if (Math.abs(diff) >= 0.0001)
                faultyBefore++;
            if (Math.abs(diff) >= 0.00001)
                faultyAfter++;
        }
        infil.close();
        System.out.println("Ej accepterade axlar ökar från " + 100* (double) faultyBefore/ totally
                           + " % till " + 100 * (double) faultyAfter/ totally + " %.");
    }//main
}//Faulty
```

b)

```
import java.io.*;
public class Faulty {
    public static void main(String[] arg) throws IOException {
        DataInputStream infil = new DataInputStream(new FileInputStream("data.bin"));
        int totally = 0;
        int faultyBefore = 0;
        int faultyAfter = 0;
        while(infil.available() > 0) {
            totally++;
            double diff = infil.readDouble();
            if (Math.abs(diff) >= 0.0001)
                faultyBefore++;
            if (Math.abs(diff) >= 0.00001)
                faultyAfter++;
        }
        infil.close();
        System.out.println("Ej accepterade axlar ökar från " + 100* (double) faultyBefore/ totally
                           + " % till " + 100 * (double) faultyAfter/ totally + " %.");
    }//main
}//Faulty
```

c) Därför att binärfiler kräver mindre minnesutrymme, samt är enklare och effektivare att bearbeta.

Uppgift 2

```
import java.util.*;
import java.io.*;
public class Transaction {
    public static void main(String[] arg)  throws IOException {
        if (arg.length < 2)
            System.out.println("Programmet fodrar två argument!");
        else {
            HashSet<String> codeSet =  new HashSet<String>();
            try {
                BufferedReader loggFil = new BufferedReader(new FileReader(arg[0]));
                try {
                    BufferedReader codeFil = new BufferedReader(new FileReader(arg[1]));
                    String code = codeFil.readLine();
                    while (code !=  null) {
                            codeSet.add(code);
                            code = codeFil.readLine();
                    }
                    codeFil.close();
                }
                catch (FileNotFoundException e) {
                    System.out.println("Filen " + arg[1] + " existerar inte");
                }
                String trans = loggFil.readLine();
                while (trans !=  null) {
                    if (codeSet .contains(trans.substring(0,7)))
                            System.out.println(trans);
                    trans = loggFil.readLine();
                }
                loggFil.close();
            }
            catch (FileNotFoundException e) {
                System.out.println("Filen " + arg[0] + " existerar inte");
            }
        }
    }//main
}// Transaction
```

Uppgift 3

a)

Klassen måste göras serialiserbar, dvs klasshuvudet skall bytas till:

```
import java.io.*;
public class Product implements Serializable {
```

b)

```
import java.io.*;
public class Order {
    public static void main(String[] arg) {
        File in = new File("lager");
        File out = new File("order");
        if (!in.exists()) {
            System.out.println("Filen lager existerar inte!");
            System.exit(0);
        }
        else if (!in.canRead()) {
            System.out.println("Filen lager får inte läsas!");
            System.exit(0);
        }
        if (out.exists()) {
            System.out.println("Filen order existerar redan!");
            System.exit(0);
        }
        ObjectInputStream infile = new ObjectInputStream(new FileInputStream(in));
        ObjectOutputStream outfile = new ObjectOutputStream(new FileOutputStream(out));
        try {
            while (true) {
                Product obj = (Product) infile.readObject();
                if (obj.getNrInStock() <= 5)
                    outfile.writeObject(obj);
            }
        }
        catch (IOException e) {
            // Nått slutet av filen
        }
        catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
        finally {
            infile.close();
            outfile.close();
        }
    }//main
}//Order
```