Program 3

Due Date: June 8, 2004 **100 Points**

This program analyzes the effects of applying the Bell-LaPadula Model's rules to a multilevel secure system.

Program Function

You are to write a program that takes a file containing a list of security levels and categories, followed by a list of subjects and objects with labels. You are to print, for each subject, a list of objects that it can read, and a list of objects that it can write, assuming the discretionary access controls allow each subject read and write access to each object. That is, you need to apply the simple security condition and *-property rules to the input.

Command Syntax

Your program must be named *mlscheck*, and must accept the following command-line syntax:

```
mlscheck filename
```

where *filename* is the name of the file to be checked. The output is to be printed on the standard output.

Input File

The input file has the following form:

```
classes     unclassified      confidential      secret      topsecret
categories  cat1              cat2              cat3
subject     A                 confidential      cat1
object      B                 secret            cat1        cat2
subject     C                 topsecret         cat1        cat3
object      D                 unclassified
object      E                 confidential      cat1
```

The `classes` keyword is followed by 1 or more words separated by blanks or tabs. These are the security classifications, and they go from lowest (on the left) to highest (on the right).

The `categories` keyword is followed by one or more words separated by blanks or tabs. These are the different categories. They are unordered, and there may be a different number of categories than classifications.

The `subject` keyword is followed by name, then a security clearance, then zero or more categories. These are separated by blanks or tabs. The subject's label is (*clearance*, { *category* … }). If there are no categories listed, the subject has an empty category set.

The `object` keyword is followed by name, then a security classification, then zero or more categories. These are separated by blanks or tabs. The subject's label is (*classification*, { *category* … }). If there are no categories listed, the object has an empty category set.

If no classes are given, or any subject's or object's class is not in the list of classes, print an error message and exit. If the categories line is omitted, there are no categories. If the categories line is present but no category identifiers are listed, there are no categories. If any subject or object names a category not in the list, print an error message and exit.

Output

The output is to look like this:

```
A can read D E
A can write B E
C can read D E
```

What To Turn In

Turn in your fully commented source code and a Makefile bundled into a *tar*(1) file. When unpacked, the *tar* file should create a directory called *pr3* that contains the program and the Makefile The Makefile should be constructed so that the grader can *cd* into the newly-created directory, type

```
make
```

and then invoke *mlscheck* as described above. If the grader has to do anything more, you ***will*** lose points!

Extra Credit

Add a command-line option –i to interpret the classes and categories as integrity clearances/classifications and categories, and apply the Biba model rather than the Bell-LaPadula model, when –i is given.