University of Toronto
Department of Computer Science
CSC 228 - Summer 1993

Assignment 2: Tracking Suspected Criminals

Due: in tutorial, June 29

This assignment will serve as an introduction to master file update, and give you a taste of the good life, writing law enforcement applications software.

The problem

You have been made responsible for maintaining a database of criminal suspects as part of an international sting operation. Law enforcement officials wish to monitor suspects and have them placed under arrest only when they become a definite public threat (eg. by killing someone). Therefore, arrests are to be made when suspects commit a *major* felony or three *minor* felonies in one year.[1]

There are six possible infractions which a suspect may commit:

1. **murder**: a major felony, punishable by 10 years in prison
2. **armed robbery**: a major felony, punishable by 5 years in prison
3. **drug trafficking**: a minor felony, punishable by 3 years in prison
4. **rioting**: a minor felony, punishable by 2 years in prison
5. **misuse of CDF computers**: a minor felony, punishable by 1 year in prison
6. **failure to look both ways before crossing the street**: a misdemeanor, punishable by a $100 fine

Note that only the first two infractions are *major* felonies. Misdemeanors should be reported, but do not influence arrests or prison terms.

In addition to the previous infractions, you may receive any of the following update requests from law enforcement officials:

1. **add suspect**: create a new record for the indicated suspect
2. **delete suspect**: remove inactive suspect record from database (this should be allowed only for suspects who have committed no felonies)

and any of the following information requests from law enforcment officials:

1. **list suspects**: display the names and current whereabouts of all suspects
2. **details suspect**: display the name and current whereabouts of the indicated suspect, and provide a criminal history including all infractions and total length of prison terms served

Once arrested, a suspect is automatically incarcerated in the federal penitentiary of his or her choice for a period of time as determined by the accumulation of all unpunished infractions. Criminals should be released from prison at the end of a sentence, automatically. However, their criminal histories should not be erased. Note, again, that suspects are only arrested following the committal of a major felony or three minor felonies in one year.

All of the infractions and law enforcement requests will be stored in a user-readable transaction file, which will be applied against the previous contents of the database whenever your program is run.

[1] This should not be taken as any reflection of reality. Furthermore, neither the sentencing scheme, nor the leniency on minor felons is intended to indicate advocacy for such a legal system on the part of the instructor nor of the University of Toronto.

Any transaction which violates a syntactic or semantic requirement should not be processed further, but reported in an error log file. Some potential violations that might occur are: unknown infraction or request type, attempt to add an already existing suspect, felony committed by unknown suspect, and attempt to delete the record of an imprisoned criminal.

To simplify timekeeping, you may assume that your program is run exactly once per year, and that all arrests and releases from prison occur simultaneously.

Files

Your database will consist of two files:

- **suspects file**: a sequential *binary* file which stores name, current whereabouts, etc. of all known suspects
- **events file**: a *binary* file which stores information about infractions and arrests in inverse chronological order linked lists, ie. most recent event first, one list per suspect

The transaction file is organized as follows:

1. law enforcement update requests and infractions (in sorted order)
2. law enforcement information requests (not necessarily in sorted order)

The first section of the transactions file is sorted by the same key as the suspects file, which uniquely identifies each suspect. You may assume that this section of the file has already been sorted when your program receives it. It is your responsibility to design the key, but some ideas are to use name, social insurance number, or interpol suspect identification number. The second section of the transactions file is not necessarily in sorted order.

Some suggestions concerning file organization will be provided in class and in the tutorials.

An example transaction file follows:

```
add             10883596  Toonces Cat   123 Maple St  Toronto  \
COMPUTER-MISUSE 12539876  Gumby Green   23/03/87               |
RIOTING         23443122  Rock  Stone   12/01/87               | for pass1
del             67549012                                       |
ROBBERY         73552156  Cliff Hanger  07/09/87               |
del             77783551                                       /
list                                          \
details         77783551                      |  for pass 2
list                                          /
```

Processing

The processing will proceed in two passes. In the first pass, you will handle *add* and *delete* requests, and process any infractions that have been recorded, using the master file update algorithm. During the first pass, you should also release any prisoners who have reached the end of their sentences.

In the second pass, you will handle the law enforcement information requests. For the *list suspects* request, you will have to read and display every record of the suspects file. For the *details suspect* request, you should try to locate the desired information *without* reading every record. To do this efficiently, you can use the binary search algorithm.

For those students wishing to be fancy, you might also allow a **change suspect** request which would modify fields such as a suspect's name or address. Such requests would naturally belong in the update requests section of the transaction file.

Developing your program

Like the first assignment, your program must be modularized. This time, however, the choice of program file names and the functions of each are completely up to you.

In addition to updating the database files, your program will produce an output log file which provides the output requested by the law enforcement officials. Your program must also produce an error log file which describes any erroneous transactions. As before, you are responsible for designing your own transactions files.

The information stored in the database is completely up to your discretion. If you wish to include additional infractions in your transaction file, or extra features in the program, make sure that these are well documented in your report.

How you will be marked

As always in this course, your tutor will look mainly at your report. She or he will also look at your program to get an indication of the quality of your program design and coding, but your mark will be based primarily on what can be learned from your report. You should, therefore, discuss clearly the data structures you used, the breakdown into separate modules, and the decisions you made about how to handle special cases.

The test input and output you provide will be very important for this assignment, so think carefully about what cases you need to check, and make sure the output is easily readable. Be careful not to provide too much test data.

Please re-read the section in the course guide dealing with how assignments will be marked, and pay careful attention to any additional instructions your tutor provides.