Coordination of Specialized Programs

Consider the following problem, which is based on an exercise by John Stone:

The file `Iowa-cities.dat` contains information about the sixty largest cities and towns in Iowa: their names and populations, as determined by the 2000, 1990, and 1980 censuses. A typical line of the file looks like this:

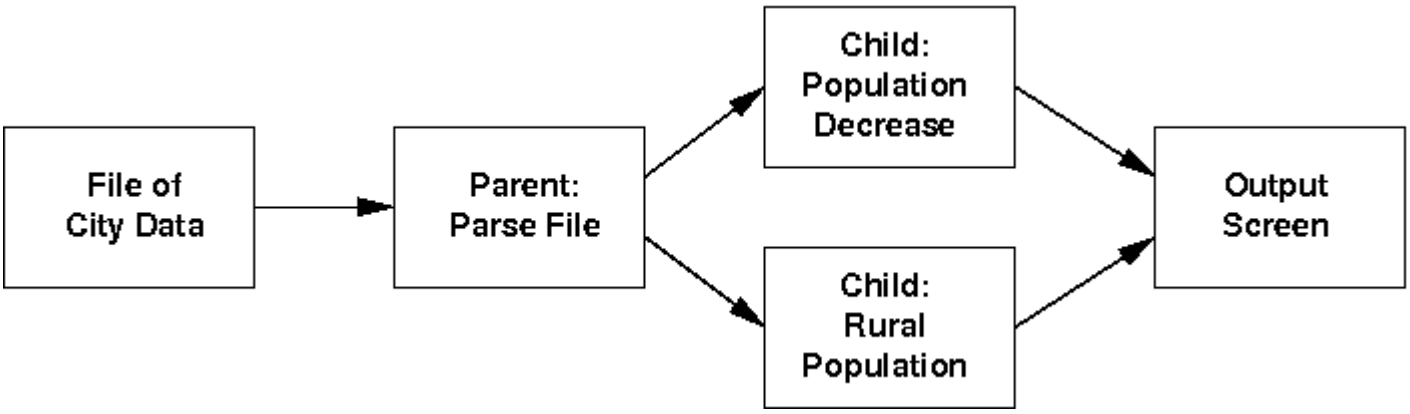
```
Grinnell,9105,8902,8868
```

There are 4 fields separated by commas. Field 1 is the name of the town (at most 16 characters); field 2 contains the 2000 population;
field 3 contains the 1990 population; field 4 contains the 1980 population.
(See `fgets(3)`, `strtok(3)` and `atoi(3)` about parsing lines of the file.)

In this part of the lab, you are to write a C program which reads data from this file and determines the answers to the following two questions:

- Which of these Iowa cities had the largest percentage decrease in population between 1980 and 2000?
- The 2000 census population of the state of Iowa was 2,926,324. How many Iowans lived *outside* the sixty largest cities and towns in 2000?

The basic approach for this problem is illustrated in the following diagram:



As this diagram suggests, one parent process will spawn two children; the parent and both children then should collaborate to perform this work. Overall, the parent process will read the file and send some data from each line through a pipe to both children. Each child will answer one of the above questions. More precisely:

- The parent process should use `pipe(2)` before `fork(2)` to open a communication channel with a child. The parent then should send formatted data from the file to that child using the pipe.
- The first child process should read from the pipe and determine the city with the greatest percentage decrease in population.
- The parent process should similarly create a second child connected to the parent with a pipe. The parent will also send the relevant data from the file through this pipe.
- The second child process should read the data from the pipe to compute the rural population directly.

Note that the child processes should not have any unnecessary file descriptors (or file streams) available in them.