the input those words to be excluded from the index listed. Consider the next page displaying an input file and the expected `concord1` output for that file. (These correspond to `in07.txt` and `out07.txt` from the test files provided to you for this assignment.)

```
1
''''
of
and
the
too
on
who
to
that
"""
that fortune
sense and sensibility
life of robert browning
the man who knew too much
legend of montrose
visit to iceland
orthodoxy
the mountains
on the track
ward of king canute
```

```
life of robert BROWNING
ward of king CANUTE
that FORTUNE
visit to ICELAND
ward of KING canute
the man who KNEW too much
LEGEND of montrose
LIFE of robert browning
the MAN who knew too much
legend of MONTROSE
the MOUNTAINS
the man who knew too MUCH
ORTHODOXY
life of ROBERT browning
SENSE and sensibility
sense and SENSIBILITY
on the TRACK
VISIT to iceland
WARD of king canute
```

The input file has a version number (for this assignment the version is 1), followed by a line with '''' (i.e four single quotes). There then follow the list of *exclusion words*, one word per line, with the list itself ended using a line with """" (i.e. four double quotes). Lastly the remainder of the file is made up of the lines-for-indexing of which the concordance will constructed for that file.

Each line in the output contains text based on one line-for-indexing of the input with exactly one word capitalized. When reading the output line-by-line (i.e., from top to the bottom) the capitalized words are in alphabetical order. Some lines-for-indexing *appear more than once in the output* as they have more than one indexed word. And as mentioned earlier in the assignment, the problem for A#1 has been simplified in that no line-number references appear in the output. (We will add this in subsequent assignments.)

Your task it to write `concord1.c` which constructs such output from such input.

Running the program

Your program will be run from the Unix command line. Input is expected from `stdin`, and output is expected at `stdout`. ***You must not provide filenames to the program, nor hardcode input and output file names.***

For example, assuming your current directory contains your executable version of `concord1.c`, (i.e., named `concord1`), and a `tests/` directory containing the assignment's test files is also in the current directory, then the command to transform the previous page's input into required output will be:

```
% cat tests/in07.txt | ./concord1
```

In the command above, output will appear on the console. You may want to capture the output to a temporary file, and then compare it with the expected output:

```
% cat tests/in07.txt | ./concord1 > temp.txt
% diff tests/out07.txt temp.txt
```

The same thing (i.e., producing output and comparing it with the expected output) can be combined into a one-liner:

```
% cat tests/in07.txt | ./concord1 | diff tests/out07.txt -
```

The ending hyphen/dash informs `diff` that it must compare the contents of `tests/out07.txt` with the input piped into `diff`'*s* stdin.

Exercises for this assignment

1. Write your program `concord1.c` program in the `a1/` directory within your git repository.
    - Read `stdin` line by line and processing each line appropriately.
    - Read and store exclusion words.
    - Read and store lines-for-indexing