1 Problem Description

Problem 1 involves creating a lexical analyzer for a custom language "Kanpur" using flex, given the specifications in the assignment PDF and on Piazza.

2 Instructions to Run

The subdirectory 'problem1' contains 3 files: prob1.l, prob1.sh and this PDF. To run, simply:

- Open the file prob1.sh. It contains a variable named 'file_name'. Rename it to the relative path of the testcase file. Note that the variable name should **not** contain the file extension as it is always assumed to be .knp. For example, if the testcase file is "private1.knp", simply rename the variable file_name to "private1". The script with handle the rest.
- Now, run prob1.sh. It can be executed in 2 ways:
  - ./prob1.sh -f: This redirects the output to the file {file_name}.output
  - ./prob1.sh: This will simply display the output to stdout.
- The script automatically deletes the extra files that it has created (a.out and lex.yy.c).

3 Corner/Error Cases and Format of Output

- The scanner scans all the tokens till it encounters EOF and reports all (if any) errors it has encountered.
- The output is sorted by lexeme.
- Since keywords are case-insensitive, they have the same counter. However, if there are occurrences in different cases, all of the cases are reported (with their total sum of appearances as the count).
- The following are error cases that are flagged by the scanner:
  - **Invalid floats**: Since floats such as .1421 are not allowed by the language prescription, they are reported as an error.
  - **Float capacity**: Floats can only have up to 6 decimal digits in the language. Thus, any float with more than 6 decimal digits is marked as an error.
  - **Hexadecimal float**: Floating-point numbers with hexadecimal are not allowed (for eg, $0x12.c3$) and are marked as an error.
  - **Leading zeros**: Leading zeros are not allowed in integers, hexadecimal or floating point numbers and are raised as an error by the scanner.
  - **Invalid identifier**: Identifiers can only begin with a letter followed by letters/digits. Thus, any identifier beginning with a letter is marked as an invalid identifier.
  - **Invalid string**: Strings enclosed in " " but having a ' in between/unclosed strings are flagged as errors. (Their equivalent versions with ' are error cases as well).
  - **Unrecognized characters**: Characters that are not accepted by the language are marked by the lexical analyzer.
- The following cases, in my opinion, were not very direct and required some thinking/modification in the code to incorporate and should be tested:
  - Common counter for keywords/operators whilst displaying them as separate rows in the output.

– Different counter for identifiers in different case. For example, xyz and xYz are different identifiers and should have separate counters.

– Strings enclosed in " " with multiple occurrences of ' in it: For example,
"This is an 'invalid' string"
This entire string should be marked as an error.