Assignment — C Messages

CS 265 Advanced Programming Techniques

Overview

Write a C *program* to verify messages. A Finite State Machine would suffice, but a parser would be fine.

To Do

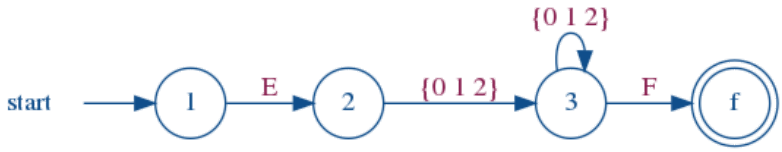
We have developed a protocol for reading messages from a device. There are different length messages. Each message has rules, described below. Messages are separated by a newline. Your job is to verify that the stream is valid, that it contains only valid messages.

Each message is followed by a description, some examples, and a *Deterministic Finite State Automaton* (*DFA*) which recognises the message. Before input is read, you are in state 1. We transition states on each character read. If, at the end of input, we are in an *accepting state* (double circle), then the message is valid.

foo:

Starts with an `E` followed by a string of digits `0` `1` or `2` followed by an `F` . E.g.:

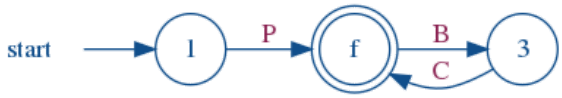
```
E201022011101F
```



eep:

Starts with a `P`, followed by arbitrary number of `BC` (including none). E.g.:

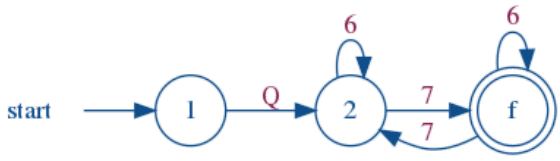
```
P
PBCBCBC
```



op:

Starts with a `Q` . Followed by a string of `6` and `7` , where the number of `7` s must be odd.

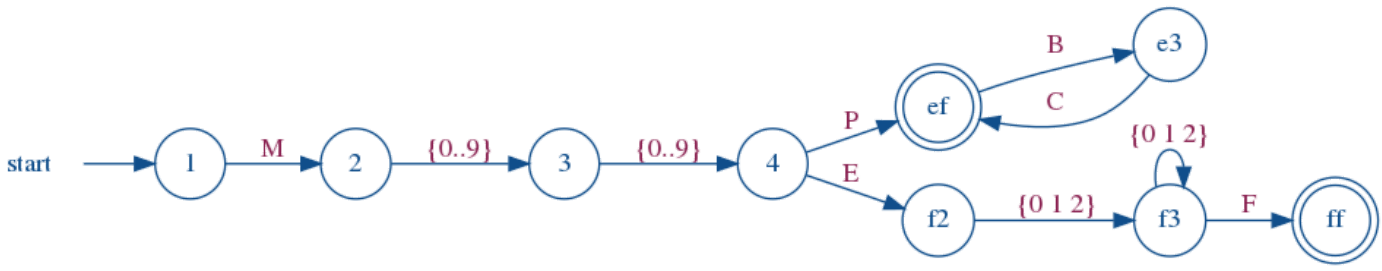
```
Q7
Q66666676666
Q76767
```



ork:

Starts with an `M`. Followed by 2 decimal digits, then a *foo* or an *eep*

```
M84E2010201F
M27PBC
```



Input

You will read input from the filename provided as the first argument on the comand line. If you can not open it for reading, print a meaningful message, and quit. If no filename is provided, you will read **`stdin`** . **Note**, **`stdin`** is of type `FILE*`, and can be read exactly like a file. See **`Labs/C/inputLines.c`** for an example.

Output

Print out the message, followed by a space, then **`OK`** if the message is valid, **`FAIL`** otherwise. One per line.

Hints

You maybe find these helpful. If you do not, that is okay. They are not directives.

- Read each line, parse the string, or
- Use `fgetc()`, parse the input directly
- Make a function for each message

Style

We've covered style. Use judicious (not excessive) whitespace, proper file header coments, function header comments, other helpful (but not effusive) comments. Apply principles you've learned, to make your code legible.

What to Submit

Submit just the source code, **`msg.c`** .

- Do not submit a .doc file, a .zip file, or a .pdf file. These formats are not correct for this assignment and will not be accepted.
- Make sure you develop and test your script on tux. If your script does not run on tux, it will not be considered correct.