Data Structure

Lab 4. Regular Expression

Shin Hong

Apr 28, 2020

Lab 4: Submission & Evaluation

- Submission deadline: 4 PM, May 1 (Fri)

- Online test: http://hisolve.handong.edu Session ID: `DTSdl4w`

- Evaluation: 5 points x 1 problem (total 5 points)
  - 5 points: succeed on time (before the deadline)
  - 4 points: succeed by 9 PM, May 4 (Mon)
  - ≤ 3 points: submit a report by 4 PM, May 5 (Tue)

- Note
  - You must use doubly linked list as a queue
    https://github.com/hongshin/DataStructure/blob/linkedlist/doubly/linkedlist.c
  - You must construct a solution as a single source code file

Regular Expression (1/2)

Regular expression (RE) is a language to specify patterns of finite sequences of characters (i.e., texts). For its intuitive syntax and rich expressiveness, RE has been used widely in various interfaces of computer systems. When we use `ls` (or `dir`), we specify a condition of file name in RE. For instance,

```
ls c?*t
```

will returns the file names that starts with `c` and ends with `t`, and contains at least one character between the first and the last characters, for example, `cat`, `cast`, `coot`, `colonist`.

In this lab, you are asked to write a program that receives a RE pattern and a string, and then determines whether or not the string matches to the given RE pattern.

Regular Expression (2/2)

Let's first define the RE language that we will use specifically.

A pattern of RE consists of a sequence of *articles*. A string is matched to an article if it satisfies a certain condition. Our RE has the following five kinds of articles:

- one character between 'a' to 'z' : it matches to the string contains the one specific character
- * : it matches to an arbitrary string including empty string
- ? : it matches to an arbitrary length-1 string
- ! : it matches to either an arbitrary length-1 string or empty string
- [$c_1c_2\cdots c_n$] where c_i, is one character between 'a' and 'z' : it matches to one of length-1 string that contains either c_1, c_2, ... or c_n.

A given string is matched to a RE pattern when the string is a concetnation of a sequence of strings that matches to the sequence of articles in order. For instance, pattern `[ab]?d*[ef]` matches to aade, bcdkkf, aaddddddddde, but does not matches to ade, cade, abbe, addk.

Input and Output

- Input
  - Given from the standard input
  - First line contains a RE pattern. The length of a pattern does not excced 128. No whitespace exists in a middle of the pattern.
  - From second to sixth lines (total 5 lines), each line has a string whose length does not exceed 128.
- Output
  - Print whether each string is mached to the given pattern in sequence.
  - Print '`True`' if a string matches; '`False`' otherwise.
  - Make sure to put newline ('`\n`') at the end

```
*is*[tu]!↲
height↲
christ↲
minimalist↲
isoaspartate↲
lister↲
```

<Input>

```
false↲
true↲
true↲
true↲
false↲
```

<Output>