Regular expressions (practices Qs with sols)

1. Give a regular expression that matches valid email addresses. A valid email address, for the purposes of this problem, consists of a string of alphanumeric characters, followed by an "@" sign, followed by a domain name that is a string of alphanumeric characters followed by ".com" or ".edu" Subdomains are allowed.

2. For strings containing the letters 'a', 'b', and 'c', give a regular expression that captures all strings that have at least two (different, consecutive) letters in alphabetical order.

3. Give a regular expression that can match street addresses. In our definition, a street address consists of a number, followed by an optional letter, for the house/apartment number, then one or more capitalized words for the street name, then either "Dr.", "Rd." or "St." for the street type.

Regular expressions (cont.)

4. With football season upon us, your boss wants you to write a program that scrapes sports websites for information about football players. Write a regular expression that captures information about a football player, in the following format:

   <First name> <Last name>, <Height>, <Weight>, <Position>

   Where first and last names start with capital letters and are followed by 0 or more other letters, height is a numeric height in feet, followed by an apostrophe, then a numeric height in inches, followed by a quotation mark, weight is a number followed by either "lb" or "kg" and position is one of the following abbreviations: QB, OL, RB, TE, WR, DL, LB, S or CB.

5. For strings containing the letters 'a', 'b', 'c', and 'd' give a regular expression that captures all strings that use their letters in *reverse alphabetical order*, but use at most three of the four possible letters (note that the strings themselves can be longer than 3 letters long, since letters can repeat).

Solutions

1. Note that there are several (equivalent) ways to write this regular expression. Here is one.

   Let $\alpha =$ [a-zA-Z0-9]

   $\alpha^{+}@(\alpha^{+}\ .)^{+}(\text{com}|\text{edu})$

2. **Answer:** We can capture this by specifying the three possible substrings that should make our regular expression match: 'ab', 'bc' and 'ac'. We need *one* of those substrings to appear somewhere in the string we are trying to match:

$$[a-c]^{*}((ab)|(bc)|(ac))[a-c]^{*}$$

3. **Answer:** There are many ways to write a regular expression to capture street addresses. Here is one possible answer, with the symbol σ representing a space:

$$[0-9]^{+}[A-Za-z]?\sigma([A-Z][a-z]^{*}\sigma)^{+}(Dr|Rd|St).$$

Solutions (cont.)

4. **Answer:** There are many possible regexes that could match the given description. Here's one:

$$[A{-}Z][a{-}z]^* \; [A{-}Z][a{-}z]^*, \; [0{-}9]'[1{-}2][0{-}9]'', \; [0{-}9]{+}(lb|kg), \; (QB|OL|RB|TE|WR|DL|LB|S|CB)$$

5. **Solution** The regular expression that captures the above language is:

$$(d^*c^*b^*)|(d^*b^*a^*)|(d^*c^*a^*)|(c^*b^*a^*)$$

Deconstructing this a bit, note that each of the four "clauses" only uses three of the four letters. Each letter can be repeated zero or more times (repeating zero times is how we can get strings that use fewer than three different letters), and the ordering ensures that the letters appear in reverse alphabetical order.