Java 4 Lab: Regular Expressions using Java due Sunday night, 10/2/22, 11:59 pm

Java has a regular expression pattern package that you can import and use. The main point of regular expressions is finding patterns in input strings.

The first demo programs will have various matching/replacing matches function and each of these functions uses two important strings: the regular expression and the input to test if there’s a match.

|  |  |  |  |
| --- | --- | --- | --- |
|  | pattern | Comments on pattern | Comments on code |
| RegexExample1.java | **.s** | Will match any target string with a single char followed by an **s** | Shows 3 different ways to determine T/F match |
| RegexExample2.java | User input | Input: target string and  regex string | Again shows 3 ways but with user regex string |
| RegexExample3a.java | **Number or Identifier** | Input: target string  Output: T/F for number or Identifier | Loops so you can try different test targets to see if number or java identifier |
| RegexExample3b.java | **Number or Identifier** | Uses \d for digit in number regex and \w for word chars in Identifier | Requires (as do prev examples) standalone patterns to recognize |
| RegexExample4.java | **Number or Identifier** | Replaces numbers with X and identifiers with ID in the string | Numbers and identifiers can be mixed in the string |

Once you have worked through the demo code (running each one and noting the regular expression used to match your input string), do the following:

1. Write a replacement regex to replace all whitespace in an input string.
2. Write a regex that matches Operators and replaces them with a common symbol (like OP). Once you test it in a separate java program, add it and your whitespace code (from (1)) to the existing replacement code in RegexExample4.java (rename Answer1.java) to be able to transform strings like x=y + 32; 🡪 ID=IDOPX;
3. Write a program that can transform legal (simple) java statements into S (Answer2.java)
4. Code Blocks:
   1. Write, on paper, the next steps to ACCEPT a block of java statements. You can assume that all java blocks have {} even though those with a single statement do not need {}.
   2. Write the regular expression code to ACCEPT a CB (Code Block) and save it as Answer3.java (you can start with Answer2.java).
5. IF/ELSE
   1. Write the pattern that corresponds to an IF/ELSE statement in java. Again, assume Code Blocks have {}
   2. Write the code that ACCEPTS a legal IF/ELSE statement. You can use Answer3.java but be careful about where you insert the new lines of code.