Lab 7 (October 2 or October 3)

Instructions: Complete the exercises below. Be sure to show your code to one of the lab TAs before you leave, so that you can receive credit for this lab. You must also upload a copy of all your source code (.java) files to the link on Blackboard by 11:59 PM on Tuesday, October 3.

1. *Run-length encoding* is a relatively simple technique for compressing data; if a particular value (or substring) appears multiple times in a row, it is only represented once, followed by an integer indicating the number of times it should be repeated. For example, the string "AAAAA" can be compressed to "A5" (5 occurrences of 'A'), saving three characters in the process.

   Define a Java method named `expand()` that takes a single `String` argument. You may assume that this `String` is run-length encoded and contains an even number of characters (each pair of characters consists of a printing character followed by a ***single*** digit representing its repetition count). Your method should return a new `String` that contains the expanded (or decoded) version of its argument. For example, calling `expand("a3b7a2c4")` would return the `String "aaabbbbbbbaacccc"`.

   Complete this problem by writing a complete Java program that prompts the user to enter a run-length encoded string, passes it to your `expand()` method, and displays the decoded result.

   Hint: Java's `Integer.parseInt()` method may be helpful here if you want to avoid performing character arithmetic.

2. On some old Unix terminals, pressing the Backspace (delete) key generates the character sequence `^H` (Control-H) to delete the previously-typed character. Some people use this for sarcastic effect when writing, to pretend that they are deleting or replacing some insulting comment:

   Be nice to this fool^H^H^H^Hgentleman, he's visiting from corporate HQ.

   Define a Java method named `backspace()` that takes a single `String` argument. This method returns a new `String` where ***each*** "^H" sequence from the argument has been removed and applied to the non-Backspace character immediately preceding the "^H". For example, given the input above, the program would produce the output:

   Be nice to this gentleman, he's visiting from corporate HQ.

   where each "^H" is missing and has effectively erased a preceding character from "fool". Your program should only perform erasures for a full "^H" character digram; other

carets (as in "2^2") should be passed through unchanged. You may assume that "^H" will never appear as the first two characters of the input. You may assume that the input will never end with a caret character.

Note: You should ***only*** "erase" upon encountering the exact character sequence "^H" (a caret followed by a capital 'H'; the substring "^h" should be passed through unchanged).

Hint: A loop (use a `while` loop rather than a `for` loop) is necessary, but the `String` method `substring()` may also be invaluable.

Complete your answer to this problem by writing a complete Java program that reads a line of text from the user then displays the result of running that input through your `backspace()` method.

Sample input: Hello, world!
Expected output: Hello, world!

Sample input: A Horse is a Horse, of course, of k^Hcourse
Expected output: A Horse is a Horse, of course, of course

Sample input: I wish^H^H^Hant some ice cream^H^H^H^H^Hmilk
Expected output: I want some ice milk

Grading Guidelines: This lab is graded on a scale of 0-3 points, assigned as follows:

0 points: Student is absent or does not appear to have completed any work for the lab

1 point: Student has written only one program, but it does not compile or run at all due to errors.

2 points: Student has written (or attempted to write) both programs, but only one compiles and runs without error.

3 points: Student has written both programs, and they both compile and run correctly, without any apparent errors.