CS310: Advanced Data Structures and Algorithmns

Spring 2014 Assignment 2

Due: Tuesday, February 11, 2013 in class

Instructions

1. **Goal of this assignment** – Practice lists, stacks, queues and basic mappings.
2. question 1 should be done in your course directory. **Open an account if you haven't yet!!!**. Do any scratch work at our site in the hw2 subdirectory of your cs310 directory.

Questions

1. Transfer the qual solution to your development machine and set up an eclipse (or DrJava) project for it, after creating a src/cs310 directory with the sources and editing in a package statement. Also create an empty classes directory before creating the project: these directories guide the IDE to setting up what we want. Build and run the qual. Transfer the result back to your cs310/hw2 directory, so that cs310/hw2/src/cs310 has the source files. Run it on UNIX. This is a dry run for project deliveries.
2. Write Java functions (static methods) that provide the following computed mappings. Do not use Maps, just very simple functions:
   (a) 'a'→ 0, 'b'→ 1, . . . ,'z'→ 25, and also 'A'→ 0, . . . 'Z'→ 25 (in one map) Note that Java supports arithmetic with char variables: ch – 'a' is 0 if char ch is a, 1 if it is b, and so on.
   (b) "aa"→ 0, "ab"→1, "ac"→2,. . . ,"az"→25, "ba"→26, "bb"→27, . . . , "zz"→(26*26-1)
   (c) The inverse of b: input a number and return a pair of letters (in other words – reverse the directions of the arrows in b).
3. Problem 6.1
4. Look up String.hashCode() in the JDK documentation. From the formula, compute the hashCode of "", "G", and "GH". Note that the integer values for character is their ASCII code.
5. We want to count the number of occurrences of each letter pair in a document. Suppose a specific pair is in variable "String curPair".
   (a) How can we count it in one call to the 2b function plus one more (fast) operation?
   (b) How would you do the same thing with a HashMap? Write a code skeleton.
6. Write a routine that uses the Collections API to print out the items in any Collections in reverse order. Do not use a ListIterator. This is a generic method, like the methods in Fig. 6.11.
7. Problem 6.2
8. (a) Suppose a List<String> list1 has elements "A", "B", "C", and "D". What is returned by:
      1. list1.iterator().next();

2. list1.listIterator().next();
3. list1.listIterator(2).next();
4. list1.listIterator(4).previous();

(b) Say what is deleted (or what happens) if next/previous is replaced by remove in all of the above operations. Explain.

(c) If we had the following sequence of commands:
list1.listIterator(2).next(); list1.listIterator(2).remove(); list1.listIterator(4).previous();
what would be returned? What would the list look like following these operations?

9. As explained on pg. 259 and also on pg. 444, you can use a Stack to check balance of parentheses and other brackets. Show the sequence of stacks and the conclusion for:

(a) {[{()()}()]}

(b) {[{()]

10. Simplify checkBalance pg. 453 to handle a String of just various bracket characters like "[{()", and use a Stack<Character> for the stack.