1 Short Answers

Complete the following problems from the book and turn them in at the beginning of class on Wednesday.

4.1
4.5
4.9
4.10
4.11

2 Lab Description

In this assignment we will try an experiment in Artificial Intelligence. The idea is to write a program which will read in text and then use that text to generate some new text. The method for generating the text uses simple probability - we read the text character by character. We keep track of how often each three-character sequence appears. From this we can compute the probability that a certain character will immediately follow two given characters. For example, if the text is: “the theatre is their thing”, e appears after th 3 times, and i appears after th 1 time; no other letters appear after th. So the probability that e follows th is .75; the probability that i follows th is .25; the probability that any other letter follows th is 0.

Once we have the text processed, and stored in a structure that allows us to check probabilities, we then pick two letters (for example, the first two in the input text) to use as a beginning for our new text. Finally we use random numbers to choose subsequent characters based on the preceding two characters and the probability information. Limit the output text to no more than 200 characters.

2.1 Goals

The goals are to become proficient in using objects to build more complex structures, to learn the importance of careful class design, and to learn the Vector, Association, and ReadStream classes.

2.2 Important Considerations

You should think about the design of this program carefully before entering the lab. What would constitute a good data structure for this problem? The table of information should support requests of the form:

- update probabilities given a new triple of characters.
- select a character given a pair of characters and a random number (float) between 0.0 and 1.0.

A 3-dimensional array might seem reasonable at first, but its size would be quite large (approximately 27,000 entries if you include blanks and punctuation). Instead I would like you to use a Table class which is implemented as a Vector of Associations. Each Association would have a 2-character pair as its key, along with a value which is a frequency list. The frequency list would keep track of which characters appeared after the given 2-character pair, along with a frequency.

Each frequency list should be an object of another class, FrequencyList, that you will define. How should the frequency list be implemented? Well, ... another Vector of Associations sounds like a good idea. Thus the frequency list’s Vector would consist of Associations in which the key was a single character and the value was a count of the number of times that letter occurred after the pair with which the list is
associated. Think carefully about what methods the frequency list needs to support and any other instance
variables that might be useful.

The data structure design built from these two classes has the benefit of having only as many entries as
necessary for the given input text.

You will find it extremely helpful to look carefully at the word frequency program on page 48 of Bailey
(or better yet, a revised version of the program in the Sample Programs folder on-line).

Warning: When you import the package with the class Random, use

    import java.util.Random;

If you write “import java.util.*;”, the program will get confused as to which version of the Vector
class it should use as there is one in java.util as well as one in structure.

2.3 Input

I suggest you first debug your program using as input a String constant (e.g., “the theater is their thing”)
until it works properly. After it works, you can take input from the keyboard using the ReadStream class.

The ReadStream class is part of the Bailey’s structure package. You can find the documentation from
the 136 web page. The methods you will have the most need for include

    char readChar();
    boolean eoln();
    boolean eof();

End of file is signalled for Java on the Mac (and, indeed, any UNIX system) by typing “Control-D” on a
new line.

Of course, you must create a new ReadStream before using any of these methods. If you use the construc-
tor with no parameters, the input will come from the window created by System.in, which is the window
in which you execute java progName.

2.4 Output

After the input has been processed you should generate a new string using the frequencies in the table. You
may start with a fixed pair of letters that appears in the table or choose one randomly. Generate and print
a string of at least 200 letters so that we can see how your program works.

2.5 Extras

There are many ways in which you might extend such a program. For example, as described, letter pairs
which never appeared in the input will never appear in the output. Is there a way you could introduce a bit
of “mutation” to allow new pairs to appear?

Instead of working at the character-pair level, you could work at the word-pair level, although this results
in huge numbers of legal word pairs, so only attempt this after you get the required work finished.

2.6 What to hand in

A complete design for the two classes is due at the beginning of lab on Wednesday. You must turn in a copy
of the design which includes a listing of instance variables for each class as well as the declaration of each
method in each of the classes (for your own benefit you should also include the bodies of these methods).
Finally you must include a description of how you will test these classes. Keep a copy of this for your own
use as it may take us a while to grade these (though we intend to get them back to you in the early part of
the lab). This time we will assign a grade (10% of the grade for this assignment) based on the correctness
and completeness of the information you hand in.

The program is due at 11:59 p.m. on Sunday, 1 March. Programs are handed in by putting a folder,
whose name includes your last name, into the CS136 drop-off folder on Cortland. The folder should contain
both the .java and .class files. As always, try to test your classes thoroughly.