Homework 5
CS5046

Due by 5:30PM, March 31, 2004
Submit by email to murali@cs.vt.edu

This homework will explore the issues we discussed in class related to inheritance. Read the material from Chapters 8-10 of Barnes and Kölling to help you with this homework. Keep in mind the principles of good design we have discussed earlier. Start with the Simple Cell project on the class web page (http://people.cs.vt.edu/~murali/teaching/cs5046/lectures/simple-cell.tgz). In this homework, you will refine this class.

1. (10 points) Add a subclass of Molecule called MRNA. The class stores a string corresponding to the sequence of mRNA molecule.

2. (25 points) Make the Molecule class an abstract class with an abstract method whose signature is "public Molecule act()." Implement the act() method in each of the concrete subclasses of Molecule. This method performs the appropriate action in each subclass of Molecule (as specified by the Central Dogma of Molecular Biology) and may return a new Molecule created by the “action.” For example, the act() method in the Gene class will transcribe the gene into an instance of MRNA and return that object. In the MRNA class, the act() method takes the mRNA sequence stored in the class and converts it into an instance of Protein using the genetic code. The act() method in Protein simply “folds” the protein and does not return any molecule.

3. (25 points) In Homework 4, you implemented a method called isValid() that takes a String as argument and returns true if and only if the string is a “valid” amino acid. In this homework, implement an equivalent of that method, with the signature "bool isValid()," for each subclass of Molecule so that a client can invoke the method as follows:

Molecule mol;
boolean check = mol.isValid();

The method in the Gene/MRNA class should simply check if the stored sequence is a valid genomic/mRNA sequence. You do not have to ensure that the mRNA sequence ends in a stop codon.

I leave the design of the isValid() methods open to you but I will penalise code duplication. You may have to consider various issues when implementing this sub-problem. Think about whether the implementations of the isValid() methods in each class are very similar. If so, should there be a class called ValidityChecker, which contains the common portions of these methods yet maintains the flexibility to address the slightly different checks that each subclass of Molecule needs? Alternately, can you make do with an implementation of isValid() in the Molecule class without over-riding implementations in the subclasses?

4. (40 points) Add a public static void main() method to the Cell class that performs the following tasks:

(a) Creates a new instance of Cell.
(b) Adds a user-specified number $k$ of instances of Gene to the Cell instance. The user must specify this number as the first argument on the command line. Assume that $k \leq 10$. Pick 10 genes that you like and obtain their IDs, symbols, and sequences from GenBank.

(c) Repeatedly loops over the Molecules stored in the Cell instance until all $k$ Genes have been translated into Proteins that are also folded. In each loop, call the act() method on each molecule with a probability of 0.5, i.e., you will decide whether to invoke the act() method or not by flipping a fair (unbiased) coin. Output the total number of times you execute the entire loop. Keep in mind that when you invoke act() on an instance of Gene, you are simulating the expression of that gene into one mRNA copy. Therefore, you should remove the Gene instance from the Cell instance and add the instance of mRNA. Removing an element of a Java collection when iterating over the collection can be a little tricky. Read the documentation for the Iterator interface in the java.util package carefully.

- Make sure you test all your solutions thoroughly. Consider all types of inputs a user may provide and how you will deal with them. Read Chapter 6 to refresh your memory about the types of tests you should perform.

- Do not forget to include full documentation for your source code as described in Chapter 5.10 of Barnes and Kölling.

- Create a folder called `<YourName>-Homework5` and use a separate folder inside this folder for the source code for each problem.

- Submit your homework by emailing the files containing the source code to me. I prefer it if you can zip the entire folder and send the zipped folder to me. If doing so is difficult, just email me the individual files as attachments.