Name: ________________________________
Login: __________
Honor code acknowledgment (signature) ________________________________

<table>
<thead>
<tr>
<th></th>
<th>value</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem 1</td>
<td>20 pts.</td>
<td></td>
</tr>
<tr>
<td>TOTAL:</td>
<td>20 pts.</td>
<td></td>
</tr>
</tbody>
</table>
PROBLEM 1:  *(Your DNA is closer than my DNA)*

Similarity of proteins can be determined by many methods. In this problem you’ll compute a simplistic similarity measure (or metric) for two proteins by simply adding up the similarity scores for the corresponding amino-acids in each protein.

For example, if the proteins have the same length (we’ll assume this to be the case) then the example below shows how the similarity score for two proteins can be determined by the similarity of the corresponding amino acids (the values for the amino acid correspondence are taken from the BLOSUM 62 substitution matrix, this isn’t what that matrix is really used for, but we’ll use it for this purpose in this problem.)

```
protein 1       R   C   Q   E   G   H
protein 2       S   C   D   Q   C   E
amino similarity -1  9  0  2  -3  0
```

The score for the proteins is $-1 + 9 + 0 + 2 + -3 + 0 = 7$.

**Part A (10 points)**

You will write a method to determine the similarity score for two proteins (amino acid sequences). You’re given a method that returns the similarity score for two amino acids (the method isn’t shown, but its specification is).

Complete method `proteinSimilarity` so that it works as specified. For example, if `ps` is a `ProteinScore` object, then the code below should print 7 as shown above.

```java
int sim = ps.proteinSimilarity("RCQEGH", "SCDQCE");
System.out.println(sim);
```

Complete the method on the next page. Note, you will need to call `aminoSimilarity` in the code you write, but you’re writing code in method `proteinSimilarity`. 
public class ProteinScore
{
    /**
     * Compare two amino acids and returns a similarity score in
     * the range -20 to 20, where a higher value means the more
     * similar the amino acid parameters are.
     * @param aa1 is one amino acid
     * @param aa2 is another amino acid
     * @return a value indicating how similar the amino acids are
     */
    public int aminoSimilarity(String aa1, String aa2) {
        // not shown, returns score based on BLOSUM 62
    }

    /**
     * Return a score for how close protein p1 is to p2
     * using the similarityScore for each of the amino acids
     * in protein p1 and protein p2 (we assume p1 and p2 have
     * the same length).
     * @param p1 is one protein
     * @param p2 is the other protein
     * @return a similarity score for the proteins
     */
    public int proteinSimilarity(String p1, String p2) {
    }
}
Part B (10 points)

Typically in some applications you’ll want to know which protein from some database or list of proteins is most similar to a given protein. Write the method `mostSimilar` (in the class `ProteinScore` as above) that returns which of many proteins stored in an array is closest to a given protein that’s a parameter. For example, if the parameter `prot` is "QEG" and the array contains "QEM", "QEK", "QEL", and "QES" then the method should return "QES" since this is the most similar.

The other methods from `ProteinScore` above aren’t shown below, but you can call `proteinSimilarity` or `aminoSimilarity` in writing `mostSimilar`.

Assume each String in the array has the same length as the String `prot`.

```java
public class ProteinScore {
    /**
     * Returns the protein (string) from plist that is most similar to parameter prot, using proteinSimilarity to determine how similar the proteins are. All strings in plist have the same length as the String prot. 
     * @param prot is protein being matched 
     * @param plist is the list of candidate proteins 
     * @return the protein in plist that is most similar to prot 
     */
    public String mostSimilar(String prot, String[] plist){
        // Code goes here
    }
}
```