**44-141 Computer Programming I**

**Programming Project 4 (20 points)**

Name your Project 4 source code file using your last name, followed by the first initial of your first name, followed by Project4.py.

For example, Joyce Smith’s file would be named SmithJProject4.py. Put this Project 4.py file into a zipped file using the same naming format except with the zip extension. Place your zipped file in the Project4 dropbox by midnight on the due date.

Also download the Project 4 Grading Sheet. Rename this file using the format SmithJProject4GradingSheet.docx, where SmithJ is replaced by your name. Type your name in the Grading Sheet and save this file. Place this grading sheet in the Project 4 dropbox also by midnight on the due date.

Programs submitted late will receive NO CREDIT.

**NOTE: You may discuss this problem with others and share ideas, but YOU MUST WRITE THE CODE YOURSELF. You may not obtain code from another person, and you may not share your code with another person.**

**The Problem**

Scientists measure how closely related a species is by looking at the DNA sequences for key proteins and seeing how similar/dissimilar they are. If the two sequences of DNA are essentially the same, the two species are considered to be evolutionarily closer since there is a relationship between changes and time. This process is called sequence alignment.

Consider the two strings of DNA below (completely made up, miss matches in red):

Species 1: AATAACGAAA

Species 2: AA**A**ACGAAAA

A scientist can change the alignment by assuming that an insertion or deletion, of one of the bases has occurred. They could make such a change, called an **indel** for short, to see if it improves the alignment:

Species 1: AATAACGAAA-

Species 2: AA-**A**ACGAAAA

Assuming two indels, marked as two dashes(-), the alignment is greatly improved. The scientist would assume that two changes happened, one change in each species.

While complex algorithms exist to do sequence alignment, it is also useful to support a researcher and allow them to do an alignment by hand.

**Your Task**

The program should do the following:

1. You will prompt for two strings. The strings can have any characters you like, but to be "biological" it should consist of: "A", "T", "C", "G". The strings **do not** have to be of the same length.
2. You will then prompt for one of 3 commands:
   1. "a" for add. Add an indel
   2. "d" for delete. Delete an indel
   3. "s" for score. Score the present alignment
   4. "q" for quit. Stop the process.
3. **Adding an Indel**. When you add an indel, you must prompt for two pieces of information:
   1. which string to change
   2. at what index (starting at 0) do you wish to place the indel (placement is **before** the given index, **Error** if the index is out of range).

The string should then be modified and a dash(-) added.

1. **Delete an Indel**: If you can add an indel, you should be able to delete it if it doesn't do what you want. Again, you must prompt for two pieces of information
   1. which string to change
   2. the index (starting at 0) to delete the indel. It is an **Error** to delete a character that is not an indel.
2. **Scoring**. You will report the number of matches and the number of mismatches.
   1. Any indel is automatically a mismatch.
   2. If one string is shorter than the other, the shorter string is filled out with indels.
   3. After you score, you print both strings.
      1. Matching characters are printed in **lower case**. If the user entered upper case letters, you convert them to lower case on a match.
      2. All mismatches are printed in **upper case**.
      3. Indels are printed as dashes.

**Additional Information:**

1. As before, try to do this in pieces:
   1. get the command loop working
   2. get individual commands to work (scoring is the most work, save it till last)
2. Printing the string is best done by creating a new string and adding characters (upper, lower or dashes) to that string and then printing it. Then clear it before the next printing.

**Sample Run (User input is shown in bold):**

Enter DNA 1: **aaabbbccc**

Enter DNA 2: **aabbbcccc**

Enter a command:

a(add indel)

d(delete indel)

s(score)

q(quit): **s**

Matches: 7 MisMatches: 2

String 1: aaAbbBccc

String 2: aaBbbCccc

Enter a command:

a(add indel)

d(delete indel)

s(score)

q(quit): **a**

Which String to work on (1 or 2): **2**

Before which index to place the indel?: **2**

String1: aaabbbccc

String2: aa-bbbcccc

Enter a command:

a(add indel)

d(delete indel)

s(score)

q(quit): **s**

Matches: 8 MisMatches: 2

String 1: aaAbbbccc-

String 2: aa-bbbcccC

Enter a command:

a(add indel)

d(delete indel)

s(score)

q(quit): **d**

Which String to work on (1 or 2): **2**

Delete what index (start at 0): **2**

String1: aaabbbccc

String2: aabbbcccc

Enter a command:

a(add indel)

d(delete indel)

s(score)

q(quit): **s**

Matches: 7 MisMatches: 2

String 1: aaAbbBccc

String 2: aaBbbCccc

Enter a command:

a(add indel)

d(delete indel)

s(score)

q(quit): **q**

**Program Documentation:**

* Include your name and course number and section in the program as comments.
* Use good programming style.
  + For example, use blank lines to separate major sections of code.
  + Use comments to indicate the major steps in your program algorithm.
  + Use back slash ( \ ) character if the statement continues onto the next line and is lengthy.

**Programming suggestions:**

* The program above is an algorithm. Follow each of the steps and find the appropriate Python statements that will accomplish each step.
* Write a small set of Python statements at a time, and then compile your Python program. This way you will be able to identify your errors as you develop your program.
* Do not add any features to the program that you turn in. Do not include Python statements that we have not studied at this point in the course.

**How to Get Help:**

If you have trouble, you can get assistance in the following ways:

* Read through the lecture notes. There might be a command to solve the problem.
* Email the Instructor or the Teaching Assistants with questions.
* Go to Teaching Assistant’s Lab hours for help. (Lab hours are posted on the course web site.)
* Set up an appointment with the Instructor or the Teaching Assistant to get one on one help.