**Date assigned:** Friday, March 11, 2016          **Date due: Friday**, March 18, 2016

Points: 40

**Goals for this assignment**

1. Read and use existing code
2. Write an object-oriented program using two classes.
3. Use composition, friends, and operator overloading.
4. Practice modular programming by using well-defined functions.

You must copy the Visual Studio solution 04_DNASet from the public folder on Grace to your desktop. Rename the folder 04_DNASet to 04_DNASet_PUNetID. The Solution name inside of Visual Studio will not change.

This Solution contains the project 04_DNA. This is a complete DNA class *similar* to the class you provided in assignment 3. You must use the provided DNA class for assignment 4. Read the header file DNA.h carefully so you understand how the class works. A simple driver, DNADriver.cpp, demonstrates how to use the class. **Do not edit DNA.h or DNA.cpp.**

You must create a new project, named 04_DNASet, in the solution. This new project must be linked to the 04_DNA project and be linked to the DNA.obj file produced by that project. All of your work for this assignment will happen in 04_DNASet.

You must provide a class, DNASet, that represents a set of DNA objects. This class will be similar to RationalSet.

You must provide the following functionality for the set:

> Add a DNA strand to the set. The set can contain a maximum of 100 DNA objects. A set cannot contain the same strand more than once.
> Get the current number of items in the set.
> Determine if a particular DNA strand is in the set.
> Provide the set_union operation.
> Provide the set_intersection operation.
> Provide a way to retrieve a specific DNA strand from the set. The user will provide an integer, 0 to the number of items in the set −1, an you will provide the correct DNA strand.
> Provide the operator>> Read up to 100 DNA objects from the stream. Each DNA object is separated by whitespace.
> Provide the operator<< Print each DNA object to the stream, separated by a *space* (do not print a newline).
> Provide an appropriate default constructor.

You may add other operations as necessary.

**Data**

You must copy two data files, Pool1.txt and Pool2.txt, from Grace to the 04_DNASet project.

**Driver**

The driver for this project must:

1. Read each file (Pool1.txt and Pool2.txt) into its own DNASet.
2. Print each set to the screen.
3. Determine which DNA strands appear in both sets and print those strands to the screen.
4. Determine the longest strand in each set and print each of those strands to the screen. In the case of a tie, use the strands closest to the top of the file.
5. Determine the one strand in Pool1 and one strand in Pool2 that are most similar to each other and print both strands to the screen. Remember, P-Stat can only handle two DNA strands of the same length. In the case of a tie, use the strands closest to the top of the file.

**Notes on design:**

You are to use object-oriented design using composition.

For the DNASet class you will need a .h and a .cpp. You will also need a driver (named 04_DNASetDriver.cpp) containing the main function.

You may not change the DNA class provided to you.

**Error Handling**:

- You should be able to handle a empty file. This is not an error; reading an empty file merely produces an empty set.
- The files do not need to contain the same number of strands or contain strands of the same length.
- The DNA class will automatically discard bad data from the file. The DNA class will read the line: AAABTTT as one strand (AAA) and discard everything from the B to the next white space.


**Sample Files** (Each file must end with a blank line)

**Pool1.txt**                **Pool2.txt**


**CAT**                      **TAC**
**AAA**                      **AAT**
**T**

```
Sample Output

********************
* DNA POOL ANALYZER *
********************

Pool 1
------
CAT AAA T

Pool 2
------
TAC AAT

In both Pools
-------------


In at least one Pool
--------------------
CAT AAA T TAC AAT

Longest Strand in each Pool
---------------------------
Pool1: CAT
Pool2: TAC

Most Similar:
-------------
P-Stat: 0.33
Pool1: CAT
Pool2: AAT
```

**To complete this assignment you must submit the following:**

1. **An electronic copy of your program on Grace**
   a. **Submit a solution folder titled 04_DNASet_PUNetID**
   b. Type your program (fully documented/commented) into the project. You need to follow the coding standards from the CS250 Web page. These coding standards have been modified to include additional C++ language features introduced in CS250, so please be sure to read the new coding standards. Make sure that you include the hours you worked on the assignment in your header comments.
   c. Pay attention to the example output. Your program's output must look **exactly** like the sample output. The spacing and newlines in your output must match exactly.

d. Make sure that your program builds without errors & warnings and runs correctly. If you get any errors or warnings, double check that you typed everything correctly. Be aware that C++ is case-sensitive. You will lose 10% if there are any warnings and 70% if your program does not build successfully.

e. Once you are sure that the program works, it is time to submit your program. You do this by logging on to Grace and placing your complete solution folder in the **CS250-0X Drop** folder.

f. The solution must be in the drop folder by the time class starts on the day the assignment is due. **NO LATE SUBMISSIONS ACCEPTED. SEE NOTE BELOW**.

2. **A hard copy of your program**
   a. The hard copy must be placed on the instructor's desk by the time class starts on the day that it is due.
   b. Do not print DNA.h or DNA.cpp or DNADriver.cpp
   c. **The hard copy must be printed in color, double-sided, and stapled in the upper left corner if your solution contains multiple pages**. Failure to print properly will result in loss of 3.5 points (10%)
   d. Your tab size must be set to 2 and you must not go past column 80 in your output.

Because this is due the day before Spring Break and because you have one class session to work on this project: **You cannot submit this project late!**

**Remember, if you have any problems, come to me straight away**

**with your project on Grace. Good Luck!!!!**