University of Wollongong
School of Computer Science and Software Engineering

CSCI124/MCS9124 Applied Programming Spring 2015

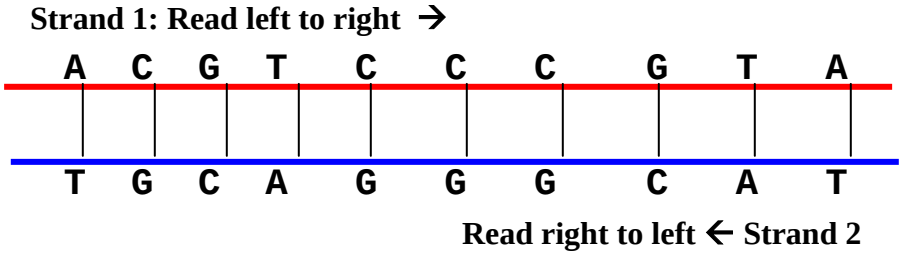
Lab 2 (1 mark)
Due at 11:59pm Thurs, 19th March.

Background:
The aim of this lab is to write a simple C++ function that tests your knowledge of file input and string manipulation, and then an exercise in using **`make`**.

Task:
Step One:
The DNA molecule is a double-helix. You can think of this double helix as two parallel sequences of DNA with nucleotides or bases (A,C,G,Ts) on one strand matched with their associated nucleotide on the other strand according to the following rules:

```
A  matches with  T
C  matches with  G
```

For example, imagine we have removed the double-helix and show the two strands in parallel; notice that every base 'A' on one strand matches a 'T' on the other strand and that every 'C' on one strand is matched with a 'G' on the other strand:



Databases that store the DNA sequence of an organism typically *only* store one of the two strands of the DNA to save room. Since the two DNA strands are complements of one another, it is "easy" to get the second complementary strand if you have the original strand. If a base on strand 1 is an 'A', then the base on the other strand must be a 'T'. Note however, that strand 1 begins at the left and is read from the left to the right: "ACGTC…" but strand #2 begins at the right and is read from the right to the left: "TACGG…". Because the strands are read in "reverse" order and they have bases that are "complements" of one another, the strands are commonly called "reverse complements".

This exercise involves writing a function for an existing C++ program, which consists of 5 files, available separately as `dna.cpp`, `dna.h`, `freq.cpp`, `freq.h` and `main.cpp`, in the zipped file `lab2.zip`. The only file you are required to change is `dna.cpp`. All the other files are to remain the same. Some of these files do nothing important – they're there just for the second part of this exercise. The only function you have to write is reverse_complement. The stub for this function can already be found in `dna.cpp`:

```
void reverse_complement(ifstream& fin)
{
}
```

This function should read the input text file into the character array located in the global area of the source file, ignoring any newlines. It should then produce the complement strand of DNA, placing it into the same array, but it will need reversing. It should finally reverse the array and print it out. For example, if the input file called `ecoli.fna` contained only the sequence

```
ACTTGGCC<newline>
GGTAC<newline>
```

the C++ program `dna`, when run as shown below, should produce the reverse complement of this sequence to standard output.

```
$ ./dna
Enter Filename to read: ecoli.fna
GTACCGGCCAAGT
Enter Filename to write: ecoli-rev.fna
File written.
```

The main program then asks for an output file to store the reverse complement strand. Notice that the reverse complement is written without any newlines. You can assume that the input file will have only uppercase letters (A,C,G, or T), and newlines.

Step Two:
Write a suitable makefile called `makefile` to build the final program dna from the five source and header files. There should be dependency lines for headers, plus creation commands for any `.o` files as well as the final executable. You should include a suitable clean target in your makefile. Test your makefile by using the shell command

```
$ touch file
```

to change the time-modified date for *file* and see what is remade. For submission you will need to add the extension `.txt` to your makefile.

Submit:

You are to submit `dna.cpp` and `makefile.txt` using

```
$ submit –c csci124 –a lab2 –u USERNAME dna.cpp makefile.txt
```

notice that you only need to submit the files dna.cpp and makefile.txt as these are the only ones which have changed.