CS1112 Discussion Exercise 9

1 Reverse complement

In the DNA double helix, two strands twist together and "face" each other. The two strands are reverse-complementary, i.e., reading one strand in reverse order and exchanging each base with its complement gives the other strand. A and T are complementary; C and G are complementary.

For example, given the DNA sequence

```
AGTAGCAT
```

the reverse sequence is

```
TACGATGA
```

so the reverse complement is

```
ATGCTACT
```

(a) Write a function `rComplement(dna)` to return the reverse complement of a DNA strand. *Use a loop* to reverse the strand—do not use vectorized code. `dna` is a vector of characters. Assume that `dna` contains only the letters 'A', 'T', 'C', and 'G'. If `dna` is the empty vector return the empty vector.

(b) Now write a function `rCompBulk(mat)` to return the reverse complements of a set of DNA strands. `mat` is a matrix of characters; each row of the matrix represents one strand of DNA (so `mat` contains only the letters 'A', 'T', 'C', and 'G'). Return a matrix the same size as `mat` such that the rth row of the returned matrix is the reverse complement of the rth strand of DNA (the rth row of `mat`). Again *use loops*—do not use vectorized code.

2 Counting a DNA pattern

Write a function `countPattern(dna,p)` to find out (and return) how many times a pattern `p` occurs in `dna`. Assume both parameters to be strings that contain the letters 'A', 'T', 'C', and 'G' only. Note that if `p` is longer than `dna`, then `p` appears in `dna` zero times. Use the built-in function `strcmp` to compare two strings. Again, use a loop to solve this problem.

3 Counting a DNA pattern—challenge edition!

Rewrite the function `countPattern(dna,p)` *without* using the `strcmp` built-in function.