Homework 8

Rohit Kamat rgk359

Problem 1 (5 points): In bioinformatics, k-mers refer to all the possible subsequences (of length k) from a read obtained through DNA sequencing. For example, if the DNA sequencing read is "ATCATCATG", then a the 3-mers in that read include "ATC" (which occurs twice), "TCA" (which occurs twice), "CAT" (occurs twice), and "ATG" (occurs once). You can read more about k-mers on Wikipedia (https://en.wikipedia.org/wiki/K-mer).

a) Write a function that takes a string of nucleotides as input and returns a **dictionary** with all 3-mers present in that string, and the number of times that each 3-mer occurs. Then use your function to find the 3-mers in the DNA sequence `my_seq1` defined below.

The output of your function should be a dictionary that is structured like this (although it will have several more entries):

```
{"ATC": 3, "ATG": 2, "TGA": 3}
```

where each key is a 3-mer itself (e.g., "ATC") and each value is the number of times that 3-mer occurs.

b) For the sequence `my_seq2` defined below, verify manually that your function generates the correct result, and explain your reasoning in 2-3 sentences.

In [1]:
```
# Find all 3-mers in these two sequences
my_seq1 = "CAGCCCAATCAGGCTCTACTGCCACTAAACTTACGCAGGATATATTTACGCCGACGTACT"
my_seq2 = "ATCATCATG"
# Your code goes here

#define nucleotide function
def nucleotides(string):
    num=len(string)
    num_range= range(len(string))
    nucleotide_list= {}

    for a in num_range:
        if a <= (num-3):
            sequence = string[a: a+3] #"sequence" takes a length of a codon,
            if sequence in nucleotide_list:
                nucleotide_list[sequence] += 1 #a 3-mer that is already in t
            else:

                nucleotide_list[sequence]=1 # a new 3-mer is added to the di

    return nucleotide_list


print("The 3-mers and the counts of 3-mers for sequence 1 are:", nucleotides
print("The 3-mers and the counts of 3-mers for sequence 2 are:", nucleotides
```

```
The 3-mers and the counts of 3-mers for sequence 1 are: {'CAG': 3, 'AGC':
1, 'GCC': 3, 'CCC': 1, 'CCA': 2, 'CAA': 1, 'AAT': 1, 'ATC': 1, 'TCA': 1,
 'AGG': 2, 'GGC': 1, 'GCT': 1, 'CTC': 1, 'TCT': 1, 'CTA': 2, 'TAC': 4, 'A
CT': 4, 'CTG': 1, 'TGC': 1, 'CAC': 1, 'TAA': 1, 'AAA': 1, 'AAC': 1, 'CT
T': 1, 'TTA': 2, 'ACG': 3, 'CGC': 2, 'GCA': 1, 'GGA': 1, 'GAT': 1, 'ATA':
2, 'TAT': 2, 'ATT': 1, 'TTT': 1, 'CCG': 1, 'CGA': 1, 'GAC': 1, 'CGT': 1,
 'GTA': 1}
The 3-mers and the counts of 3-mers for sequence 2 are: {'ATC': 2, 'TCA':
2, 'CAT': 2, 'ATG': 1}
```

When I put the sequence of my_seq2 into the function, the function is able to take the sequence, find the length and range of the sequence, and within the sequence find all the 3-mer's and put the sequences in the dictionary and returns the dictionary. Then the second if statement determines if a certain 3-mer has been repeated or not in the list, and collects how many times a certain 3-mer has been counted within myseq2, which is collected under the dictionary "nucleotide_list" and is returned.

Problem 2 (5 points): DNA sequences are typically stored in a format called FASTA (pronounced fast-ay). A single FASTA file may contain many different sequences. For example, you may have a FASTA file for a mouse, and each mouse gene sequence is stored as a separate sequence in that FASTA file. All sequences in a FASTA file begin on a new line with a greater-than symbol ">" (without quotes).

Write a function that takes the *name* of a FASTA file as input, opens that file, counts the number of sequences in the file (by counting the number of lines in the file that start with a “>” symbol), and returns the count. Download the file "CD4.fasta (http://wilkelab.org/classes/SDS348/2017_spring/homeworks/CD4.fasta)" to your computer and use your function to count the number of sequences in the file. The file `CD4.fasta` contains amino acid sequences of the CD4 membrane protein that is found on the surface of the immune cells.

In [2]:

```
#count the sequence function
def num_sequence(s):
    counts={}
    for a in s:
        if a in counts:
            counts[a]+=1
        else:
            counts[a]=1
    return counts
#Open the file for reading and count the letters
with open("CD4.fasta", "r") as infile:
    counts= num_sequence(infile.read())

for a in [">"]:
    print("The number of sequences is", counts[a])
```

```
The number of sequences is 18
```