Bioinformatics Scientist Data Challenge

Instructions:

Write a python script that computes the following information:

1. Total number of 25mers
2. Number of distinct 25mers
3. K-mer with the highest count

Step 1 - Data Importation & Libraries

In [1]:
```
import pandas as pd
```

Read in fastA file:

In [2]:
```
openFA = open('SRR1748776.fa')
readFA = openFA.read()
```

Step 2 - Data Formatting

Split text into individual sequences and store in a list

In [3]:
```
record = []

for line in readFA.rstrip().split(">"):
    seq = line.rstrip().split("\n")
    record.append(seq)
#record
```

Working out how to extract and categorize data from the first group and how to concatenate the subsequences into a full string ...

In [4]:
```
record[1][0].split(' ')
```

Out[4]:
```
['SRR1748776.1', '1', 'length=251']
```

In [5]:
```
record[1][1] + record[1][2] + record[1][3] + record[1][4]
```

Out[5]:
```
'CGGTTCAGCAGGAATGCCGAGATCGGAAGAGCGGTTCAGCAGGAATGCCGAGACCGGATAGCGATCTCGT
ATGCCGTCTTCTGCTTGAAAAAAAAAGACAAGGCTCCTGAATTCGCGTCTGCATATCGGGTGACCATCCCC
CAAGGCCTAATCCGCCAACCTGACCGACAGCGATCCATTACCGCGAGGGAAAGGCGCTACTACCCCCTGTG
AGGTCAGCGAACCAGATCCTTACACCGGATCGGTATAGC'
```

For loop that gathers all necessary data (name, id, length, and full sequence) and stores each variable in a separate list.

In [6]:
```
names = []
ids = []
lengths = []
sequences = []

for i in range(1, len(record)) :
    name, id, length = record[i][0].split(' ')

    seq = record[i][1] + record[i][2] + record[i][3] + record[i][4]

    names.append(name)
    ids.append(id)
    lengths.append(length)
    sequences.append(seq)
```

Step 3 - Create Dataframe with all sequence information

In [7]:
```
df = pd.DataFrame(list(zip(names, ids, lengths, sequences)), columns = [
'name', 'id', 'length', 'sequence'])

df.head()
```

Out[7]:

	name	id	length	sequence
0	SRR1748776.1	1	length=251	CGGTTCAGCAGGAATGCCGAGATCGGAAGAGCGGTTCAGCAGGAAT...
1	SRR1748776.1	1	length=251	CGGCATTCCTGCTGAACCGAGATCGGAAGAGCGTCGTGTAGGGAAA...
2	SRR1748776.2	2	length=251	CGGAAGAGCGGTTCAGCAGGAATGCCGAGATCGGAAGAGCGGTTCA...
3	SRR1748776.2	2	length=251	CGGCATTCCTGCTGAACCGCTCTTCCGAGATCGGAAGAGCGTCGTG...
4	SRR1748776.3	3	length=251	CGGAAGAGCGGTTCAGCAGGAATGCCGAGATCGGAAGAGCGGTTCA...

Step 4 - Create a Dictionary that stores sequence name as key and kmer count as value

1. Creates empty dictionary kmer_count

1. Creates for loop that loops over the entire length of ~ 315,000 sequences

   k: the number of nucleotide bases in each subsequence

   seq: specifies which of the ~ 315,000 sequences is currently being evaluated

1. Creates loop that loops over the total length of the selected sequence, minus k (the number of bases in a subsequence to spare for the end of the sequence, plus 1 (to accommodate 0 index)

   kmer: the selected sequence at position i to i + k -- this local variable updates one position forward at each i to span the whole length of the sequence in groups of 25

   kmer_count[kmer]: using dict.get(key, default_return_val) method, the input string is searched for in the dictionary and a count of 1 is added if it is found. The default value is 0, meaning if the sequence is not found a zero value will be returned.

   kmer_counts: returns a list with all dictionary keys and values SORTED kmer_counts: sorts kmer_counts from highest to lowest count value using key = lambda tup: tup[1]

In [8]:
```
kmer_count = {}

for j in range(len(df['sequence'])) :

    k = 25
    seq = df['sequence'][j]

    for i in range(len(seq) - k + 1) :

        kmer = seq[i : i + k]
        kmer_count[kmer] = kmer_count.get(kmer,0) + 1

kmer_counts = kmer_count.items()
kmer_counts = sorted(kmer_counts, reverse = True, key = lambda tup: tup[
1])
```

Display kmer_counts: list of tuples

In [9]:
```
#kmer_counts
```

Step 5 - Create a Dataframe with all 25mer sequences and counts

This dataframe lists each unique 25mer sequence along with the number of times the sequence appeared throughout the entire fastA file of sequences.

In [10]:
```
counts_df = pd.DataFrame(kmer_counts, columns = ['Sequence', 'Counts'])
counts_df.head(15)
```

Out[10]:

	Sequence	Counts
0	CGGAAGAGCGGTTCAGCAGGAATGC	62777
1	GGAAGAGCGGTTCAGCAGGAATGCC	62513
2	GAAGAGCGGTTCAGCAGGAATGCCG	62472
3	AAGAGCGGTTCAGCAGGAATGCCGA	62426
4	GAGCGGTTCAGCAGGAATGCCGAGA	62328
5	AGAGCGGTTCAGCAGGAATGCCGAG	62323
6	GATCGGAAGAGCGGTTCAGCAGGAA	54693
7	TCGGAAGAGCGGTTCAGCAGGAATG	54661
8	ATCGGAAGAGCGGTTCAGCAGGAAT	54558
9	CGGTTCAGCAGGAATGCCGAGACCG	53348
10	AGCGGTTCAGCAGGAATGCCGAGAC	53345
11	TCAGCAGGAATGCCGAGACCGGATA	53333
12	GCGGTTCAGCAGGAATGCCGAGACC	53291
13	CAGCAGGAATGCCGAGACCGGATAG	53275
14	AGCAGGAATGCCGAGACCGGATAGC	53269

ANSWERS

Total Number of 25mers

In [11]:
```
print('The total number of 25mers is:', sum(counts_df['Counts']))
```

```
The total number of 25mers is: 29513632
```

Number of Distinct 25mers

Distinct k-mers should be count of k-mers that occur at least once in reads/data

In [12]:
```
print('The total number of distinct 25mers is:', len(counts_df['Counts'
]))
```

```
The total number of distinct 25mers is: 21418789
```

K-mer With the Highest Count

In [13]:
```
counts_df[list(counts_df.Counts == max(counts_df.Counts))]
```

Out[13]:

	Sequence	Counts
0	CGGAAGAGCGGTTCAGCAGGAATGC	62777

In [ ]:
```

```