Find Unique

- Deliverables:
  - findUnique.py - 50 total points
  - required input from STDIN
  - required output to STDOUT
  - output must be sorted by tRNA header

Sets and tRNA

Mitochondrial tRNA are encoded in the mitochondrial genome and account for ~10% of the coding space, yet over 50% of mitochondrial genomic disease have their origin in this molecule. These molecules are transcribed, processed, modified, and ultimately folded to produce these mature adapters between the mitochondrial transcription and translation processes. Defects in mt.tRNA maturation reflect additional disease states .. if only we could identify those mutations. We are working on such a device, though we first need to use it to identify abundance of these molecules in a mixed population. Are there any unique subsequences among the 22 mt.tRNA that can be used as "tags"? If so, we can count those tags to assess abundance.

Assignment

Write a python command line program that reads a file of fasta sequences from STDIN, finds the unique subsequences that occur in a single tRNA such that no members of this set occur among any of the other tRNA sets. Each of those 22 sets should be minimized, such that no member of that tRNA set is a substring of any other member of this set. [Unique and Essential]

As an example, let's say that both ACG and AAACGA are in a set. Since ACG is a substring of AAACGA we would remove AAACGA. [ACG is Essential]

Use Python sets for this assignment. Not only will your code be smaller, but it will be more likely to work. The union, intersection and difference operators will be quite useful.

Rough design plan...

1) compute the set of all substrings from each tRNA sequence. [powerset]

2) for each tRNAset, compute the union of all other tRNA sets, and then remove that set from the current tRNAset. Notice that this operation finds all of the other elements from all other tRNA. If any of those are present in your current tRNA, then they are not unique ! [Unique]

3) for each tRNAset, it now contains the truly unique ones, along with any extensions of that subsequence. If, for example, it was found that G only occurred in a single tRNA, then adding an A onto that G must also be unique because it has a G in it. We only want the minimal form.. G. [Essential]

4) Remove spaces from the header line before sorting and printing. This will make your output a little prettier.

Report

Print a report that contains items as follows.

- Line 1: the tRNA name
- Line 2: the tRNA sequence
- lines 3-80 or so, each unique element.

These unique elements need to be ordered by their starting position in the tRNA sequence, and should be spaced over so they are directly under that subsequence in the original tRNA. This looks like an alignment, but you can find where it belongs by using the string.find() method. Include dots in all positions to the left, serving as alignment guides for your reader. [see sample output below]

Do this for all of the 22 tRNA sequences.

Print the tRNA out as above, in sort order.

Hints:

use sets !

your final code will be under 100 lines.

Do most of your coding using class methods.

The sequences include characters that are just alignment characters. They are not part of the sequence and must be removed. [-_.] are alignment characters.

When removing items from a set, don't do this while iterating through that set. Also, when building a unique set you will need the original contents of all other sets. So.. build a new set to keep the unique contents. Notice that you build the union of all other tRNA, and this happens 22 times - these unions are all distinct from each other. Example, consider 4 sets, A,B,C,D. we would compute the set B, C, D to use against set A, and we would compute the set A, C, D to use against B. A and B need to not change while we are doing this computation.

This is a command line program, though it does not have any arguments. You really don't need the commandline class. Your input comes from STDIN and output goes to STDOUT. Use the FastaReader for input and print statement for output. If you are going to use jupyter to develop your code, you can add a filename to fastareader for your testing.

Submission

Submit code using Canvas. As always, you can work with a partner, but you are to write your own code.

In [1]:
```
#!/usr/bin/env python3
```

```
# Name: Your full name (CATS account username)
# Group Members: List full names (CATS usernames) or “None”

import sys
class FastAreader :

    def __init__ (self, fname=''):
        '''contructor: saves attribute fname '''
        self.fname = fname

    def doOpen (self):
        if self.fname is '':
            return sys.stdin
        else:
            return open(self.fname)

    def readFasta (self):

        header = ''
        sequence = ''

        with self.doOpen() as fileH:

            header = ''
            sequence = ''

            # skip to first fasta header
            line = fileH.readline()
            while not line.startswith('>') :
                line = fileH.readline()
            header = line[1:].rstrip()

            for line in fileH:
                if line.startswith ('>'):
                    yield header,sequence
                    header = line[1:].rstrip()
                    sequence = ''
                else :
                    sequence += ''.join(line.rstrip().split()).upper()

        yield header,sequence

##############################################################################
##
# Main
# Here is the main program
#
##############################################################################
##

def main(inCL=None):
    ''' '''
    pass
```

```
if __name__ == "__main__":
    main()
```

Sample Output

In [ ]:

```
tRNA|Lys|∃UU|Bostaurus|mitochondrial
CACUAAGA"LCUAUAUAGCACPAACCU∃UU6AGUUAGAGAUUGAGAGCCAU"UACUCUCCUUGGUGACCA
CACU
.ACUA
...UAA
....AAG
.......A"
.........L
..........CUAU
............AUAU
..............AUAG
................UAGC
..................GCA
....................P
......................AAC
........................ACCU
..........................∃
............................6
..............................AGU
................................GUU
..................................UUA
....................................UAGA
......................................AGAGA
........................................GAU
..........................................AUU
............................................UUGA
..............................................UGAG
................................................GAGAG
....................................................GAGC
......................................................GCC
........................................................CAU
..........................................................U"
............................................................"U
..............................................................UAC
................................................................ACUC
..................................................................UCU
....................................................................UCC
......................................................................CUU
........................................................................GG
..........................................................................GUG
............................................................................GAC
..............................................................................ACCA
```