DNA to Amino Acid Sequence using Python

```
## Importing DNA data
From NCBI website, go to Nucleotide, and then write: NM_207618.2
- Results for Mus musculus vomeronasal 1 receptor, D18 (V1rd18), mRNA
- Click in FASTA
- Copy sequence:
GGTCAGAAAAAGCCCTCTCCATGTCTACTCACGATACATCCCTGAAAACCACTGAGGAAGTGGCTTTTCA
GATCATCTTGCTTTGCCAGTTTGGGGTTGGGACTTTTGCCAATGTATTTCTCTTTGTCTATAATTTCTCT
CCAATCTCGACTGGTTCTAAACAGAGGCCCAGACAAGTGATTTTAAGACACATGGCTGTGGCCAATGCCT
TAACTCTCTTCCTCACTATATTTCCAAACAACATGATGACTTTTGCTCCAATTATTCCTCAAACTGACCT
CAAATGTAAATTAGAATTCTTCACTCGCCTCGTGGCAAGAAGCACAAACTTGTGTTCAACTTGTGTTCTG
AGTATCCATCAGTTTGTCACACTTGTTCCTGTTAATTCAGGTAAAGGAATACTCAGAGCAAGTGTCACAA
ACATGGCAAGTTATTCTTGTTACAGTTGTTGGTTCTTCAGTGTCTTAAATAACATCTACATTCCAATTAA
GGTCACTGGTCCACAGTTAACAGACAATAACAATAACTCTAAAAGCAAGTTGTTCTGTTCCACTTCTGAT
TTCAGTGTAGGCATTGTCTTCTTGAGGTTTGCCCATGATGCCACATTCATGAGCATCATGGTCTGGACCA
GTGTCTCCATGGTACTTCTCCTCCATAGACATTGTCAGAGAATGCAGTACATATTCACTCTCAATCAGGA
CCCCAGGGGCCAAGCAGAGACCACAGCAACCCATACTATCCTGATGCTGGTAGTCACATTTGTTGGCTTT
TATCTTCTAAGTCTTATTTGTATCATCTTTTACACCTATTTTATATATTCTCATCATTCCCTGAGGCATT
GCAATGACATTTTGGTTTCGGGTTTCCCTACAATTTCTCCTTTACTGTTGACCTTCAGAGACCCTAAGGG
TCCTTGTTCTGTGTTCTTCAACTGTTGAAAGCCAGAGTCACTAAAAATGCCAAACACAGAAGACAGCTTT
GCTAATACCATTAAATACTTTATTCCATAAATATGTTTTTAAAAGCTTGTATGAACAAGGTATGGTGCTC
ACTGCTATACTTATAAAAGAGTAAGGTTATAATCACTTGTTGATATGAAAAGATTTCTGGTTGGAATCTG
ATTGAAACAGTGAGTTATTCACCACCCTCCATTCTCT

- Save it as txt file, for example: dna.txt, in a folder:
DNA_translation
```

To clean the data: eliminate '' and other possible extra characters

In [1]:
```
inputfile = 'dna.txt'
f = open(inputfile, "r")
dna = f.read()

dna = dna.replace("\n", "")

dna = dna.replace("\r", "")
```

In [2]:
```
print (dna)
```

```
GGTCAGAAAAAGCCCTCTCCATGTCTACTCACGATACATCCCTGAAAACCACTGAGGAAGTGGCTTTTCA
GATCATCTTGCTTTGCCAGTTTGGGGTTGGACTTTTGCCAATGTATTTCTCTTTGTCTATAATTTCTCT
CCAATCTCGACTGGTTCTAAACAGAGGCCCAGACAAGTGATTTTAAGACACATGGCTGTGGCCAATGCCT
TAACTCTCTTCCTCACTATATTTCCAAACAACATGATGACTTTTGCTCCAATTATTCCTCAAACTGACCT
CAAATGTAAATTAGAATTCTTCACTCGCCTCGTGGCAAGAAGCACAAACTTGTGTTCAACTTGTGTTCTG
AGTATCCATCAGTTTGTCACACTTGTTCCTGTTAATTCAGGTAAAGGAATACTCAGAGCAAGTGTCACAA
ACATGGCAAGTTATTCTTGTTACAGTTGTTGGTTCTTCAGTGTCTTAAATAACATCTACATTCCAATTAA
GGTCACTGGTCCACAGTTAACAGACAATAACAATAACTCTAAAAGCAAGTTGTTCTGTTCCACTTCTGAT
TTCAGTGTAGGCATTGTCTTCTTGAGGTTTGCCCATGATGCCACATTCATGAGCATCATGGTCTGGACCA
GTGTCTCCATGGTACTTCTCCTCCATAGACATTGTCAGAGAATGCAGTACATATTCACTCTCAATCAGGA
CCCCAGGGGCCAAGCAGAGACCACAGCAACCCATACTATCCTGATGCTGGTAGTCACATTTGTTGGCTTT
TATCTTCTAAGTCTTATTTGTATCATCTTTTACACCTATTTTATATATTCTCATCATTCCCTGAGGCATT
GCAATGACATTTTGGTTTCGGGTTTCCCTACAATTTCTCCTTTACTGTTGACCTTCAGAGACCCTAAGGG
TCCTTGTTCTGTGTTCTTCAACTGTTGAAAGCCAGAGTCACTAAAAATGCCAAACACAGAAGACAGCTTT
GCTAATACCATTAAATACTTTATTCCATAAATATGTTTTTAAAAGCTTGTATGAACAAGGTATGGTGCTC
ACTGCTATACTTATAAAAGAGTAAGGTTATAATCACTTGTTGATATGAAAAGATTTCTGGTTGGAATCTG
ATTGAAACAGTGAGTTATTCACCACCCTCCATTCTCT
```

Using libraries (Key string to Values) will connect DNA sequence of 3 nucleotides with Amino Acids. For example, nucleotides 'ACA' (codon) corresponds to Amino Acid 'T'

In [3]:
```
table = {
'ATA':'I', 'ATC':'I', 'ATT':'I', 'ATG':'M',
'ACA':'T', 'ACC':'T', 'ACG':'T', 'ACT':'T',
'AAC':'N', 'AAT':'N', 'AAA':'K', 'AAG':'K',
'AGC':'S', 'AGT':'S', 'AGA':'R', 'AGG':'R',
'CTA':'L', 'CTC':'L', 'CTG':'L', 'CTT':'L',
'CCA':'P', 'CCC':'P', 'CCG':'P', 'CCT':'P',
'CAC':'H', 'CAT':'H', 'CAA':'Q', 'CAG':'Q',
'CGA':'R', 'CGC':'R', 'CGG':'R', 'CGT':'R',
'GTA':'V', 'GTC':'V', 'GTG':'V', 'GTT':'V',
'GCA':'A', 'GCC':'A', 'GCG':'A', 'GCT':'A',
'GAC':'D', 'GAT':'D', 'GAA':'E', 'GAG':'E',
'GGA':'G', 'GGC':'G', 'GGG':'G', 'GGT':'G',
'TCA':'S', 'TCC':'S', 'TCG':'S', 'TCT':'S',
'TTC':'F', 'TTT':'F', 'TTA':'L', 'TTG':'L',
'TAC':'Y', 'TAT':'Y', 'TAA':'_', 'TAG':'_',
'TGC':'C', 'TGT':'C', 'TGA':'_', 'TGG':'W'}
```

In [4]:
```
# To verify 'table' is working, run something like
table ['GGA']
```

Out[4]:
```
'G'
```

Create a function to do the translation

In [5]:
```
def translate (dna):
    """Translate a string containing a nucleotide sequence into a string
    containing the corresponding sequence of amino acids. Nucleotides ar
    tranlated in triplets using the table dictionary; each amino acid
    is encoded with a string of length 1."""

    table = {
    'ATA':'I', 'ATC':'I', 'ATT':'I', 'ATG':'M',
    'ACA':'T', 'ACC':'T', 'ACG':'T', 'ACT':'T',
    'AAC':'N', 'AAT':'N', 'AAA':'K', 'AAG':'K',
    'AGC':'S', 'AGT':'S', 'AGA':'R', 'AGG':'R',
    'CTA':'L', 'CTC':'L', 'CTG':'L', 'CTT':'L',
    'CCA':'P', 'CCC':'P', 'CCG':'P', 'CCT':'P',
    'CAC':'H', 'CAT':'H', 'CAA':'Q', 'CAG':'Q',
    'CGA':'R', 'CGC':'R', 'CGG':'R', 'CGT':'R',
    'GTA':'V', 'GTC':'V', 'GTG':'V', 'GTT':'V',
    'GCA':'A', 'GCC':'A', 'GCG':'A', 'GCT':'A',
    'GAC':'D', 'GAT':'D', 'GAA':'E', 'GAG':'E',
    'GGA':'G', 'GGC':'G', 'GGG':'G', 'GGT':'G',
    'TCA':'S', 'TCC':'S', 'TCG':'S', 'TCT':'S',
    'TTC':'F', 'TTT':'F', 'TTA':'L', 'TTG':'L',
    'TAC':'Y', 'TAT':'Y', 'TAA':'_', 'TAG':'_',
    'TGC':'C', 'TGT':'C', 'TGA':'_', 'TGG':'W'}

    protein = ""
# Check the sequence length is divisible by 3
    if len(dna) % 3 == 0:
# Loop  over the sequence
        for i in range (0, len(dna), 3):
# extract a single codon
            codon = dna [i : i+3]
# look up the codon and store the result
            protein += table[codon]

    return protein
```

Applying the translate function to 'dna' for the index between 20:935

In [6]:
```
translate (dna[20:935])
```

Out[6]:
```
'MSTHDTSLKTTEEVAFQIILLCQFGVGTFANVFLFVYNFSPISTGSKQRPRQVILRHMAVANALTLFLT
IFPNNMMTFAPIIPQTDLKCKLEFFTRLVARSTNLCSTCVLSIHQFVTLVPVNSGKGILRASVTNMASYS
CYSCWFFSVLNNIYIPIKVTGPQLTDNNNNSKSKLFCSTSDFSVGIVFLRFAHDATFMSIMVWTSVSMVL
LLHRHCQRMQYIFTLNQDPRGQAETTATHTILMLVVTFVGFYLLSLICIIFYTYFIYSHHSLRHCNDILV
SGFPTISPLLLTFRDPKGPCSVFFNC'
```

### To verify if the Amino Acid sequence is the same as in NCBI

In the same web page used before, click in CDS (under Features). Note: They specify the values 21...938 which in python is 20:938, also the last 3 are a natural stop codon not included in the downloaded protein. The values used are 20:935

- Copy the values from: /translation = " "
- Save those as a text file in the same folder (DNA_translation) created before as 'protein.txt'

In [7]:
```
# Bring the protein values to python
def read_seq(inputfile):

    with open (inputfile, 'r') as f:
        seq = f.read()
    seq = seq.replace ('\n', '')
    seq = seq.replace ('\r', '')
    return seq
```

In [8]:
```
prt = read_seq('protein.txt')
```

In [9]:
```
prt
```

Out[9]:
```
'MSTHDTSLKTTEEVAFQIILLCQFGVGTFANVFLFVYNFSPISTGSKQRPRQVILRHMAVANALTLFLT
IFPNNMMTFAPIIPQTDLKCKLEFFTRLVARSTNLCSTCVLSIHQFVTLVPVNSGKGILRASVTNMASYS
CYSCWFFSVLNNIYIPIKVTGPQLTDNNNNSKSKLFCSTSDFSVGIVFLRFAHDATFMSIMVWTSVSMVL
LLHRHCQRMQYIFTLNQDPRGQAETTATHTILMLVVTFVGFYLLSLICIIFYTYFIYSHHSLRHCNDILV
SGFPTISPLLLTFRDPKGPCSVFFNC'
```

In [10]:
```
# Comparing the results between 'prt' (from the NCBI) and results using
prt == translate (dna[20:935])
```

Out[10]:
```
True
```