# Find ORF and Translate to Protein

**Authors:** Tony Kabilan Okeke, Ifeanyi Osuchukwu
**Template Author:** Ahmet Sacan
**Date:** 01.08.2022

In [ ]:
```
# Import packages and functions
from dnatools import seq_transcribe, seq_findgene, pprint
from urllib.request import urlretrieve
from Bio.SeqIO import parse

# Download data from NCBI for testing
url = "https://www.ncbi.nlm.nih.gov/sviewer/viewer.cgi?tool=portal&save=file&log$=seqview&db=nuccore&report=fasta&sort=&id=5
urlretrieve(url, "NC_000011.fasta");
```

## Test Cases for `seq_transcribe`

In [ ]:
```
pprint( seq_transcribe('CTTACCTCAT') )
```

```
{
  "noncode": "ATGAGGTAAG",
  "mrna": "CUUACCUCAU",
  "ptn": "LTS"
}
```

In [ ]:
```
pprint( seq_transcribe('ATGAGGTAAG') )
```

```
{
  "noncode": "CTTACCTCAT",
  "mrna": "AUGAGGUAAG",
  "ptn": "MR*"
}
```

In [ ]:
```
pprint( seq_transcribe('ACGTGAATCGATAATA') )
```

```
{
  "noncode": "TATTATCGATTCACGT",
  "mrna": "ACGUGAAUCGAUAAUA",
  "ptn": "T*IDN"
}
```

In [ ]:
```
pprint( seq_transcribe('TGA') )
```

```
{
  "noncode": "TCA",
  "mrna": "UGA",
  "ptn": "*"
}
```

In [ ]:
```
# Test it on a real DNA sequence
# NC_000011.10:c5227071-5225466 Homo sapiens chromosome 11, GRCh38.p13 Primary Assembly

# Parse data downloaded from NCBI
dna = ''.join([str(seq.seq) for seq in parse("NC_000011.fasta", "fasta")])

pprint( seq_transcribe(dna) )
```

```
{
  "noncode": "GCAATGAAAATAAATGTTTTTTATTAGGCAGAATCCAGATGCTCAAGGCCCTTCATAATATCCCCCAGTTTAGTAGTTGGACTTAGGGAACAAAGGAACCTTTAATAGAA
ATTGGACAGCAAGAAAGCGAGCTTAGTGATACTTGTGGGCCAGGGCATTAGCCACACCAGCCACCACTTTCTGATAGGCAGCCTGCACTGGTGGGGTGAATTCTTTGCCAAAGTGATGGGCCAG
CACACAGACCAGCACGTTGCCCAGGAGCTGTGGGAGGAAGATAAGAGGTATGAACATGATTAGCAAAAGGGCCTAGCTTGGACTCAGAATAATCCAGCCTTATCCCAACCATAAAATAAAAGCA
GAATGGTAGCTGGATTGTAGCTGCTATTAGCAATATGAAACCTCTTACATCAGTTACAATTTATATGCAGAAATATTTATATGCAGAGATATTGCTATTGCCTTAACCCAGAAATTATCACTGT
TATTCTTTAGAATGGTGCAAAGAGGCATGATACATTGTATCATTATTGCCCTGAAAGAAAGAGATTAGGGAAAGTATTAGAAATAAGATAAACAAAAAAGTATATTAAAAGAAGAAAGCATTTT
TTAAAATTACAAATGCAAAATTACCCTGATTTGGTCAATATGTGTACACATATTAAAACATTACACTTTAACCCATAAATATGTATAATGATTATGTATCAATTAAAAATAAAAGAAAATAAAG
TAGGGAGATTATGAATATGCAAATAAGCACACATATATTCCAAATAGTAATGTACTAGGCAGACTGTGTAAAGTTTTTTTTTAAGTTACTTAATGTATCTCAGAGATATTTCCTTTTGTTATAC
ACAATGTTAAGGCATTAAGTATAATAGTAAAAATTGCGGAGAAGAAAAAAAAAGAAAGCAAGAATTAAACAAAAGAAAACAATTGTTATGAACAGCAAATAAAAGAAACTAAAACGATCCTGAG
ACTTCCACACTGATGCAATCATTCGTCTGTTTCCCATTCTAAACTGTACCCTGTTACTTATCCCCTTCCTATGACATGAACTTAACCATAGAAAAGAAGGGGAAAGAAAACATCAAGCGTCCCA
TAGACTCACCCTGAAGTTCTCAGGATCCACGTGCAGCTTGTCACAGTGCAGCTCACTCAGTGTGGCAAAGGTGCCCTTGAGGTTGTCCAGGTGAGCCAGGCCATCACTAAAGGCACCGAGCACT
TTCTTGCCATGAGCCTTCACCTTAGGGTTGCCCATAACAGCATCAGGAGTGGACAGATCCCCAAAGGACTCAAAGAACCTCTGGGTCCAAGGGTAGACCACCAGCAGCCTAAGGGTGGGAAAAT
AGACCAATAGGCAGAGAGAGTCAGTGCCTATCAGAAACCCAAGAGTCTTCTCTGTCTCCACATGCCCAGTTTCTATTGGTCTCCTTAAACCTGTCTTGTAACCTTGATACCAACCTGCCCAGGG
CCTCACCACCAACTTCATCCACGTTCACCTTGCCCCACAGGGCAGTAACGGCAGACTTCTCCTCAGGAGTCAGATGCACCATGGTGTCTGTTTGAGGTTGCTAGTGAACACAGTTGTGTCAGAA
GCAAATGT",
  "mrna": "ACAUUUGCUUCUGACACAACUGUGUUCACUAGCAACCUCAAACAGACACCAUGGUGCAUCUGACUCCUGAGGAGAAGUCUGCCGUUACUGCCCUGUGGGGCAAGGUGAACGUG
GAUGAAGUUGGUGGUGAGGCCCUGGGCAGGUUGGUAUCAAGGUUACAAGACAGGUUUAAGGAGACCAAUAGAAACUGGGCAUGUGGAGACAGAGAAGACUCUUGGGUUUCUGAUAGGCACUGAC
UCUCUCUGCCUAUUGGUCUAUUUUCCCACCCUUAGGCUGCUGGUGGUCUACCCUUGGACCCAGAGGUUCUUUGAGUCCUUUGGGGAUCUGUCCACUCCUGAUGCUGUUAUGGGCAACCCUAAGG
UGAAGGCUCAUGGCAAGAAAGUGCUCGGUGCCUUUAGUGAUGGCCUGGCUCACCUGGACAACCUCAAGGGCACCUUUGCCACACUGAGUGAGCUGCACUGUGACAAGCUGCACGUGGAUCCUGA
GAACUUCAGGGUGAGUCUAUGGGACGCUUGAUGUUUUCUUUCCCCUUCUUUUCUAUGGUUAAGUUCAUGUCAUAGGAAGGGGAUAAGUAACAGGGUACAGUUUAGAAUGGGAAACAGACGAAUG
AUUGCAUCAGUGUGGAAGUCUCAGGAUCGUUUUAGUUUCUUUUAUUUGCUGUUCAUAACAAUUGUUUUCUUUUGUUUAAUUCUUGCUUUCUUUUUUUUUCUUCUCCGCAAUUUUUACUAUUAUA
CUUAAUGCCUUAACAUUGUGUAUAACAAAAGGAAAUAUCUCUGAGAUACAUUAAGUAACUUAAAAAAAAACUUUACACAGUCUGCCUAGUACAUUACUAUUUGGAAUAUAUGUGUGCUUAUUUG
CAUAUUCAUAAUCUCCCUACUUUAUUUUCUUUUAUUUUUAAUUGAUACAUAAUCAUUAUACAUAUUUAUGGGUUAAAGUGUAAUGUUUUAAUAUGUGUACACAUAUUGACCAAAUCAGGGUAAU
UUUGCAUUUGUAAUUUUAAAAAAUGCUUUCUUCUUUUAAUAUACUUUUUGUUUAUCUUAUUUCUAAUACUUUCCCUAAUCUCUUUCUUUCAGGGCAAUAAUGAUACAAUGUAUCAUGCCUCUU
UGCACCAUUCUAAAGAAUAACAGUGAUAAUUUCUGGGUUAAGGCAAUAGCAAUAUCUCUGCAUAUAAAUAUUUCUGCAUAUAAAUUGUAACUGAUGUAAGAGGUUUCAUAUUGCUAAUAGCAGC
UACAAUCCAGCUACCAUUCUGCUUUUAUUUUAUGGUUGGGAUAAGGCUGGAUUAUUCUGAGUCCAAGCUAGGCCCUUUUGCUAAUCAUGUUCAUACCUCUUAUCUUCCUCCCACAGCUCCUGGG
CAACGUGCUGGUCUGUGUGCUGGCCCAUCACUUUGGCAAAGAAUUCACCCCACCAGUGCAGGCUGCCUAUCAGAAAGUGGUGGCUGGUGUGGCUAAUGCCCUGGCCCACAAGUAUCACUAAGCU
CGCUUUCUUGCUGUCCAAUUUCUAUUAAAGGUUCCUUUGUUCCCUAAGUCCAACUACUAAACUGGGGGAUAUUAUGAAGGGCCUUGAGCAUCUGGAUUCUGCCUAAUAAAAAACAUUUAUUUUC
AUUGC",
  "ptn": "TFASDTTVFTSNLKQTPWCI*LLRRSLPLLPCGAR*TWMKLVVRPWAGWYQGYKTGLRRPIETGHVETEKTLGFLIGTDSLCLLVYFPTLRLLVVYPWTQRFFESFGDLSTPDA
VMGNPKVKAHGKKVLGAFSDGLAHLDNLKGTFATLSELHCDKLHVDPENFRVSLWDA*CFLSPSFLWLSSCHRKGISNRVQFRMGNRRMIASVWKSQDRFSFFYLLFITIVFFCLILAFFFLL
RNFYYYT*CLNIVYNKRKYL*DTLSNLKKNFTQSA*YITIWNICVLICIFIISLLYFLLFLIDT*SLYIFMG*SVMF*YVYTY*PNQGNFAFVILKNAFFF*YTFLFILFLILSLISFFQGNND
TMYHASLHHSKE*Q**FLG*GNSNISAYKYFCI*IVTDVRGFILLIAATIQLPFCFYFMVGIRLDYSESKLGPFANHVHTSYLPPTAPGQRAGLCAGPSLWQRIHPTSAGCLSESGGWCG*CPG
PQVSLSSLSCCPISIKGSFVP*VQLLNWGIL*RALSIWILPNKKHLFSL"
}
```


In [ ]:
```
# Add your own test case that encodes 4 Proline residues, 4 Tyrosine residues
# followed by a stop codon
pprint( seq_transcribe('CCCCCCCCCCCCTATTATTATTATTAG') )
```


```
{
  "noncode": "CTAATAATAATAATAGGGGGGGGGGGG",
  "mrna": "CCCCCCCCCCCCUAUUAUUAUUAUUAG",
  "ptn": "PPPPYYYY*"
}
```


## Test Cases for `seq_findgene`

In [ ]:
```
# With a single start and a single stop codon
seq_findgene('CTTACCTCAT')
```

Out[ ]:
```
'MR*'
```

In [ ]:
```
# With a single start and a single stop codon (Complement of the previous sequence)
seq_findgene('ATGAGGTAAG')
```

Out[ ]:
```
'MR*'
```

In [ ]:
```
# With no start or stop codon
seq_findgene('ACGTGAATCGATAATA')
```

Out[ ]:
```
''
```

In [ ]:
```
# With multiple start and stop codons
seq_findgene('CCCATGGGCAACTAGTATGCCGTGA')
```

Out[ ]:
```
'MGN*'
```

In [ ]:
```
# Test it on a real DNA sequence
dna = ''.join([str(seq.seq) for seq in parse("NC_000011.fasta", "fasta")])
seq_findgene(dna)
```

Out[ ]:
```
'MKLVVRPWAGWYQGYKTGLRRPIETGHVETEKTLGFLIGTDSLCLLVYFPTLRLLVVYPWTQRFFESFGDLSTPDAVMGNPKVKAHGKKVLGAFSDGLAHLDNLKGTFATLSELHCDKLHVDP
ENFRVSLWDA*'
```

In [ ]:
```
# Add your own test case that encodes: 1 Methionine, 4 Proline, and 4 Tyrosine amino acids, followed
# by a stop codon. Add at least one nucleotide before and after this open reading frame -- make sure
# the additional nucleotides you add do not end up producing a longer ORF.
seq_findgene('CATGCCCCCCCCCCCCTACTACTACTACTGAC')
```

Out[ ]:
```
'MPPPPYYYY*'
```