Kanza Batool Haider
Independent Course Study
Supervisor: Prof. Mehmet Baysan
Date:2/7/2018

REPORT WEEK 1:

As a starter to the project thesis, to learn the concepts of sequencing is essential before I move to the development stage. Hence, I devoted my efforts in the first week for the literature review and learned the important concepts of sequencing which shall be useful while applying the concepts later in my project.

The two main concepts learned are:

- ➔ Translation of Protein Sequencing and
- ➔ Concept of Global Alignment

In the following report I have discussed the techniques to perform sequencing of the above mentioned two important concepts also with a practical implementation on sample bare sequences using python. These bits of implementation shall add upon to make the final annotation tool destined in the final project thesis.

Kanza Batool Haider
Independent Course Study
Supervisor: Prof. Mehmet Baysan
Date:2/7/2018

TRANSLATING PROTEIN SEQUENCE

Translating protein sequence means converting a given nucleotide DNA/RNA sequence to a protein sequence. The chart below represents the particular amino acid corresponding to particular codon. ***Methionine*** is the amino acid with the start codon of ***ATG*** while ***TAA, TAG, TGA*** are the stop codon**.** While translating for protein sequence, a search is run where the start codon (ATG) is spotted till the stopping codons (TAA, TAG, TGA). This chain of nucleotides from start codon to stop codon makes a protein sequence of amino acids.

One letter code	*Possible codons*	*Three letter code*	*Amino acid*
A	*GCA, GCC, GCG, GCT*	*Ala*	*Alanine*
B	*AAC, AAT, GAC, GAT*	*Asx*	*Asparagine or Aspartic acid*
C	*TGC, TGT*	*Cys*	*Cysteine*
D	*GAC, GAT*	*Asp*	*Aspartic acid*
E	*GAA, GAG*	*Glu*	*Glutamic acid*
F	*TTC, TTT*	*Phe*	*Phenylalanine*
G	*GGA, GGC, GGG, GGT*	*Gly*	*Glycine*
H	*CAC, CAT*	*His*	*Histidine*
I	*ATA, ATC, ATT*	*Ile*	*Isoleucine*
K	*AAA, AAG*	*Lys*	*Lysine*
L	*CTA, CTC, CTG, CTT, TTA, TTG*	*Leu*	*Leucine*
M	***ATG***	***Met***	***Methionine (start codon)***
N	*AAC, AAT*	*Asn*	*Asparagine*
P	*CCA, CCC, CCG, CCT*	*Pro*	*Proline*
Q	*CAA, CAG*	*Gln*	*Glutamine*
R	*AGA, AGG, CGA, CGA, CGC, CGG,*	*Arg*	*Arginine*

	CGT		
S	*AGC, AGT, TCA, TCC, TCG, TCT*	*Ser*	*Serine*
T	*ACA, ACC, ACG, ACT*	*Thr*	*Threonine*
V	*GTA, GTC, GTG, GTT*	*Val*	*Valine*
W	*TGG*	*Trp*	*Tryptophan*
X	*NNN*	*X*	*Any codon*
Y	*TAC, TAT*	*Tyr*	*Tyrosine*
Z	*CAA, CAG, GAA, GAG*	*Glx*	*Glutamine or Glutamic acid*
***	***TAA, TAG, TGA***	***	***Stop codon***

Technique:

For any given bare sequence, we need to break it in 6 ORF (Open Reading Frames)

ORF1 >> starting search from the 'very first' character of the sequence
ORF2>> starting search from the 'second' character of the sequence
ORF3>> starting search from the 'third' character of the sequence
ORF4>> complement the sequence and reverse search from 'very last' character of the seq
ORF5>> complement the seq and reverse search from 'second last' character of the seq
ORF6>> complement the seq and reverse search from 'third last' character of the seq

Search:
'ATG as the starting codon of the protein which ends by either of any stop codons i.e 'TAA' , 'TGA', or 'TAG'T

Complement phenomena >> A --->T
C --->G

Out of the 6ORFs, the one which displays the *longest ORF* is DNA's actual protein determining frame sequence

Kanza Batool Haider
Independent Course Study
Supervisor: Prof. Mehmet Baysan
Date:2/7/2018

Sample Sequence:

```
GGTTCCGTGATGCACAGCTCCTTGGTTTTAATGAGTGTTTGTGAATGCAGTTGGTGAAGAACTCAGGCGA
GCAGAGGCAATTGTGGACACCCCTACAGAAACGTCCTATACCCATGTGGCAATGCTCTGAAGAATAGCAG
GGACCTCAGGAATGTCTATGGCCATACAATGACTAAACCAAATTCCCTCATCTTCTACTGTATCATTGTT
TTAGGACTGACACTTATGAAAATCCAATTATCTGAGGAATGTGAGCTTATCATAAAGAGGCCAAACGCAA
ACCTTACCAGAGTGCCCAAGGACCTACCCTTGCAAACAACTACTTTAGATCTATCACAAAACAATATATC
TGAGCTTCAGACTTCTGACATCCTCTCATTGTCCAAGCTGAGGGTCCTGATAATGTCCTACAACAGACTC
CAGTATCTTAATATCAGTGTTTTCAAATTCAACACAGAGCTGGAATATTTGGATTTGTCCCACAATGAGC
TAAAGGTGATCTTGTGCCACCCAACAGTCAGCCTCAAGCATTTGGACCTCTCCTTTAATGCCTTTGATGC
CCTGCCTATATGCAAAGAATTTGGCAACATGTCCCAACTACAGTTCCTGGGGTTGAGCGGTTCTCGGGTA
CAAAGTTCAAGTGTGCAGCTGATTGCTCATTTGAACATCAGTAAGGTTTTGCTGGTGTTAGGAGATGCTT
ATGGGGAAAAAGAAGACCCCGAATCTCTTCGGCACGTTAGCACTGAGACTCTGCATATTGTTTTCCCGTC
GAAAAGAGAATTCCGTTTTCTTCTGGATGTGTCCGTCAGCACTACGATCGGTTTGGAACTGTCTAACATC
AAGTGTGTGCTTGAAGACCAGGGCTGCTCTTATTTCTTACGTGCTTTGTCAAAGCTTGGAAAGAATCTGA
AGCTCTCAAATCTTACCCTGAACAATGTGGAAACAACGTGGAATTCCTTCATTAATATCCTCCAGATAGT
TTGGCATACGCCAGTCAAATATTTCTCAATTTCAAATGTGAAGCTACAAGGTCAACTTGCCTTCAGGATG
TTCAATTATTCTGACACTTCTCTGAAGGCTTTGTCGATACATCAAGTTGTCACTGATGTCTTCAGCTTCC
CCCAAAGTTACATATACAGTATCTTTGCCAATATGAACATCCAAAACTTTACAATGTCTGGAACACACAT
GGTCCACATGCTGTGCCCGTCCCAAGTTAGCCCATTTCTGCATGTGGACTTTACAGATAACCTTTTAACA
GACATGGTTTTTAAAGACTGTAGAAACTTAGTTAGATTGAAAACACTTAGTTTACAAAAGAATCAGTTAA
AAAACCTTGAGAATATAATCCTCACATCTGCAAAGATGACATCCCTACAAAAACTAGACATTAGCCAGAA
TTCTCTAAGGTACAGCGATGGGGGAATCCCATGCGCCTGGACCCAGAGTTTGTTAGTTTTAAATTTGTCT
TCGAATATGCTTACAGGCTCTGTCTTCAGATGCTTACCTCCCAAAGTCAAGGTCCTTGACCTTCACAACA
ACAGGATAATGAGCATCCCTAAAGATGTCACCCACCTGCAGGCTTTGCAGGAACTCAATGTAGCATCCAA
CTCCTTAACTGACCTTCCTGGGTGCGGGGCCTTCAGCAGCCTTTCTGTGCTGGTCATCGACCATAACTCA
GTTTCCCATCCCTCTGAGGATTTCTTCCAGAGCTGTCAGAATATTAGATCCCTAACAGCGGGAAACAACC
CATTCCAATGCACATGTGAGCTGAGGGACTTTGTCAAGAACATAGGCTGGGTAGCAAGAGAAGTGGTGGA
GGGCTGGCCTGACTCTTACAGGTGTGACTACCCAGAAAGCTCTAGGGGAACTGCACTGAGGGACTTCCAC
ATGTCTCCACTATCCTGTGATACTGTTCTGCTGACTGTCACCATCGGGGCCACTATGCTGGTGCTGGCTG
TCACTGGGGCTTTCCTCTGTCTCTACTTTGACCTGCCCTGGTATGTGAGGATGCTGTGTCAGTGGACACA
GACCAGGCACAGGGCCAGGCACATCCCCTTAGAGGAACTCCAGAGAAACCTCCAGTTCCATGCTTTTGTC
TCATACAGTGGGCATGATTCTGCCTGGGTGAAGAACGAATTACTACCCAACCTAGAGAAAGATGACATCC
AGATTTGCCTCCATGAGAGGAACTTTGTCCCTGGCAAGAGCATTGTGGAGAACATCATCAATTTCATTGA
GAAGAGTTACAAGTCCATCTTTGTGCTGTCTCCCCACTTCATCCAGAGTGAGTGGTGTCATTATGAACTC
TATTTTGCCCATCACAATCTCTTCCATGAAGGCTCTGATAACTTAATCCTCATCTTGCTGGCACCCATTC
CCCAGTACTCCATCCCTACCAATTACCACAAGCTCAAAACTCTCATGTCACGAAGGACCTATCTGGAATG
GCCCACAGAGAAGAACAAGCATGGACTTTTTTGGGCAAACCTAAGAGCATCCATTAATGTTAAGCTGGTT
AACCAGGCAGAAGGAACGTGTTACACACAGCAATAAGAATATCCACC
```

Implementation Code:

```
from Bio.Seq import Seq

#bring dna sequence into our main code to keep things clean
dna                                                                                              =
"GGTTCCGTGATGCACAGCTCCTTGGTTTTAATGAGTGTTTGTGAATGCAGTTGGTGAAGAACTCAGGCGAGCAGAGGCAATTGTGGACACC
CCTACAGAAACGTCCTATACCCATGTGGCAATGCTCTGAAGAATAGCAGGGACCTCAGGAATGTCTATGGCCATACAATGACTAAACCAAAT
TCCCTCATCTTCTACTGTATCATTGTTTTAGGACTGACACTTATGAAAATCCAATTATCTGAGGAATGTGAGCTTATCATAAAGAGGCCAAA
```

```
CGCAAACCTTACCAGAGTGCCCAAGGACCTACCCTTGCAAACAACTACTTTAGATCTATCACAAAACAATATATCTGAGCTTCAGACTTCTG
ACATCCTCTCATTGTCCAAGCTGAGGGTCCTGATAATGTCCTACAACAGACTCCAGTATCTTAATATCAGTGTTTTCAAATTCAACACAGAG
CTGGAATATTTGGATTTGTCCCACAATGAGCTAAAGGTGATCTTGTGCCACCCAACAGTCAGCCTCAAGCATTTGGACCTCTCCTTTAATGC
CTTTGATGCCCTGCCTATATGCAAAGAATTTGGCAACATGTCCCAACTACAGTTCCTGGGGTTGAGCGGTTCTCGGGTACAAAGTTCAAGTG
TGCAGCTGATTGCTCATTTGAACATCAGTAAGGTTTTGCTGGTGTTAGGAGATGCTTATGGGGAAAAAGAAGACCCCGAATCTCTTCGGCAC
GTTAGCACTGAGACTCTGCATATTGTTTTCCCGTCGAAAAGAGAATTCCGTTTTCTTCTGGATGTGTCCGTCAGCACTACGATCGGTTTGGA
ACTGTCTAACATCAAGTGTGTGCTTGAAGACCAGGGCTGCTCTTATTTCTTACGTGCTTTGTCAAAGCTTGGAAAGAATCTGAAGCTCTCAA
ATCTTACCCTGAACAATGTGGAAACAACGTGGAATTCCTTCATTAATATCCTCCAGATAGTTTGGCATACGCCAGTCAAATATTTCTCAATT
TCAAATGTGAAGCTACAAGGTCAACTTGCCTTCAGGATGTTCAATTATTCTGACACTTCTCTGAAGGCTTTGTCGATACATCAAGTTGTCAC
TGATGTCTTCAGCTTCCCCCAAAGTTACATATACAGTATCTTTGCCAATATGAACATCCAAAACTTTACAATGTCTGGAACACACATGGTCC
ACATGCTGTGCCCGTCCCAAGTTAGCCCATTTCTGCATGTGGACTTTACAGATAACCTTTTAACAGACATGGTTTTTAAAGACTGTAGAAAC
TTAGTTAGATTGAAAACACTTAGTTTACAAAAGAATCAGTTAAAAAACCTTGAGAATATAATCCTCACATCTGCAAAGATGACATCCCTACA
AAAACTAGACATTAGCCAGAATTCTCTAAGGTACAGCGATGGGGGAATCCCATGCGCCTGGACCCAGAGTTTGTTAGTTTTAAATTTGTCTT
CGAATATGCTTACAGGCTCTGTCTTCAGATGCTTACCTCCCAAAGTCAAGGTCCTTGACCTTCACAACAACAGGATAATGAGCATCCCTAAA
GATGTCACCCACCTGCAGGCTTTGCAGGAACTCAATGTAGCATCCAACTCCTTAACTGACCTTCCTGGGTGCGGGGCCTTCAGCAGCCTTTC
TGTGCTGGTCATCGACCATAACTCAGTTTCCCATCCCTCTGAGGATTTCTTCCAGAGCTGTCAGAATATTAGATCCCTAACAGCGGGAAACA
ACCCATTCCAATGCACATGTGAGCTGAGGGACTTTGTCAAGAACATAGGCTGGGTAGCAAGAGAAGTGGTGGAGGGCTGGCCTGACTCTTAC
AGGTGTGACTACCCAGAAAGCTCTAGGGGAACTGCACTGAGGGACTTCCACATGTCTCCACTATCCTGTGATACTGTTCTGCTGACTGTCAC
CATCGGGGCCACTATGCTGGTGCTGGCTGTCACTGGGGCTTTCCTCTGTCTCTACTTTGACCTGCCCTGGTATGTGAGGATGCTGTGTCAGT
GGACACAGACCAGGCACAGGGCCAGGCACATCCCCTTAGAGGAACTCCAGAGAAACCTCCAGTTCCATGCTTTTGTCTCATACAGTGGGCAT
GATTCTGCCTGGGTGAAGAACGAATTACTACCCAACCTAGAGAAAGATGACATCCAGATTTGCCTCCATGAGAGGAACTTTGTCCCTGGCAA
GAGCATTGTGGAGAACATCATCAATTTCATTGAGAAGAGTTACAAGTCCATCTTTGTGCTGTCTCCCCACTTCATCCAGAGTGAGTGGTGTC
ATTATGAACTCTATTTTGCCCATCACAATCTCTTCCATGAAGGCTCTGATAACTTAATCCTCATCTTGCTGGCACCCATTCCCCAGTACTCC
ATCCCTACCAATTACCACAAGCTCAAAACTCTCATGTCACGAAGGACCTATCTGGAATGGCCCACAGAGAAGAACAAGCATGGACTTTTTTG
GGCAAACCTAAGAGCATCCATTAATGTTAAGCTGGTTAACCAGGCAGAAGGAACGTGTTACACACAGCAATAAGAATATCCACC"

def main():

       coding_dna = Seq(dna)
       orf1 = coding_dna.translate()
       print ("**********************ORF1***********************")
       print (orf1)
       print ("\n\n")

       coding_dna = Seq(dna[1:])
       orf2 = coding_dna.translate()
       print ("**********************ORF2***********************")
       print (orf2)
       print ("\n\n")

       coding_dna = Seq(dna[2:])
       orf3 = coding_dna.translate()
       print ("**********************ORF3***********************")
       print (orf3)
       print ("\n\n")

       coding_dna = Seq(dna)
       coding_dna_reverse_comp = coding_dna.reverse_complement()
       orf4 = coding_dna_reverse_comp.translate()
       print ("**********************ORF4***********************")
```

```
        print (orf4)
        print ("\n\n")

        coding_dna = Seq(dna[:-1])
        coding_dna_reverse_comp = coding_dna.reverse_complement()
        orf5 = coding_dna_reverse_comp.translate()
        print ("**********************ORF5***********************")
        print (orf5)
        print ("\n\n")

        coding_dna = Seq(dna[:-2])
        coding_dna_reverse_comp = coding_dna.reverse_complement()
        orf6 = coding_dna_reverse_comp.translate()
        print ("**********************ORF6***********************")
        print (orf6)
        print ("\n\n")




main()
```

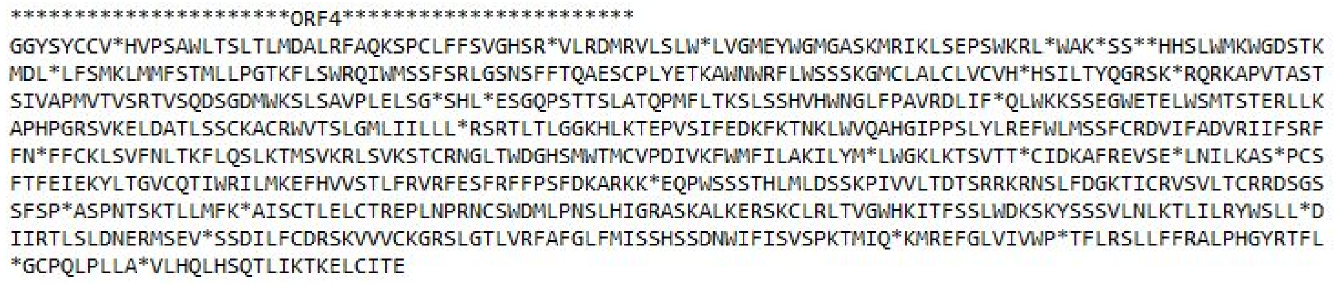
Kanza Batool Haider
Independent Course Study
Supervisor: Prof. Mehmet Baysan
Date:2/7/2018

Result

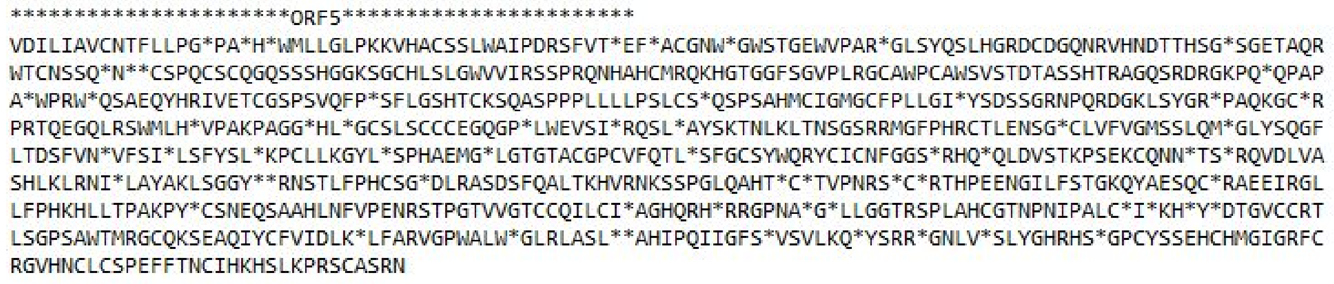
```
*********************ORF1**********************
GSVMHSSLVLMSVCECSW*RTQASRGNCGHPYRNVLYPCGNALKNSRDLRNVYGHTMTKPNSLIFYCIIVLGLTLMKIQLSEECELIIKRPNANLTRVPKDLP
LQTTTLDLSQNNISELQTSDILSLSKLRVLIMSYNRLQYLNISVFKFNTELEYLDLSHNELKVILCHPTVSLKHLDLSFNAFDALPICKEFGNMSQLQFLGLS
GSRVQSSSVQLIAHLNISKVLLLVLGDAYGEKEDPESLRHVSTETLHIVFPSKREFRFLLDVSVSTTIGLELSNIKCVLEDQGCSYFLRALSKLGKNLKLSNLT
LNNVETTWNSFINILQIVWHTPVKYFSISNVKLQGQLAFRMFNYSDTSLKALSIHQVVTDVFSFPQSYIYSIFANMNIQNFTMSGTHMVHMLCPSQVSPFLHV
DFTDNLLTDMVFKDCRNLVRLKTLSLQKNQLKNLENIILTSAKMTSLQKLDISQNSLRYSDGGIPCAWTQSLLVLNLSSNMLTGSVFRCLPPKVKVLDLHNNR
IMSIPKDVTHLQALQELNVASNSLTDLPGCGAFSSLSVLVIDHNSVSHPSEDFFQSCQNIRSLTAGNNPFQCTCELRDFVKNIGWVAREVVEGWPDSYRCDYP
ESSRGTALRDFHMSPLSCDTVLLTVTIGATMLVLAVTGAFLCLYFDLPWYVRMLCQWTQTRHRARHIPLEELQRNLQFHAFVSYSGHDSAWVKNELLPNLEKD
DIQICLHERNFVPGKSIVENIINFIEKSYKSIFVLSPHFIQSEWCHYELYFAHHNLFHEGSDNLILILLAPIPQYSIPTNYHKLKTLMSRRTYLEWPTEKNKH
GLFWANLRASINVKLVNQAEGTCYTQQ*EYP
```

```
*********************ORF2**********************
VP*CTAPWF**VFVNAVGEELRRAEAIVDTPTETSYTHVAML*RIAGTSGMSMAIQ*LNQIPSSSTVSLF*D*HL*KSNYLRNVSLS*RGQTQTLPECPRTYP
CKQLL*IYHKTIYLSFRLLTSSHCPS*GS**CPTTDSSILISVFSNSTQSWNIWICPTMS*R*SCATQQSASSIWTSPLMPLMPCLYAKNLATCPNYSSWG*A
VLGYKVQVCS*LLI*TSVRFCWC*EMLMGKKKTPNLFGTLALRLCILFSRRKENSVFFWMCPSALRSVWNCLTSSVCLKTRAALISYVLCQSLERI*SSQILP
*TMWKQRGIPSLISSR*FGIRQSNISQFQM*SYKVNLPSGCSIILTLL*RLCRYIKLSLMSSASPKVTYTVSLPI*TSKTLQCLEHTWSTCCARPKLAHFCMW
TLQITF*QTWFLKTVET*LD*KHLVYKRIS*KTLRI*SSHLQR*HPYKN*TLARIL*GTAMGESHAPGPRVC*F*ICLRICLQALSSDAYLPKSRSLTFTTTG
**ASLKMSPTCRLCRNSM*HPTP*LTFLGAGPSAAFLCWSSTITQFPIPLRISSRAVRILDP*QRETTHSNAHVS*GTLSRT*AG*QEKWWRAGLTLTGVTTQ
KALGELH*GTSTCLHYPVILFC*LSPSGPLCWCWLSLGLSSVSTLTCPGM*GCCVSGHRPGTGPGTSP*RNSRETSSSMLLSHTVGMILPG*RTNYYPT*RKM
TSRFASMRGTLSLARALWRTSSISLRRVTSPSLCCLPTSSRVSGVIMNSILPITISSMKALIT*SSSCWHPFPSTPSLPITTSSKLSCHEGPIWNGPQRRTSM
DFFGQT*EHPLMLSWLTRQKERVTHSNKNIH
```

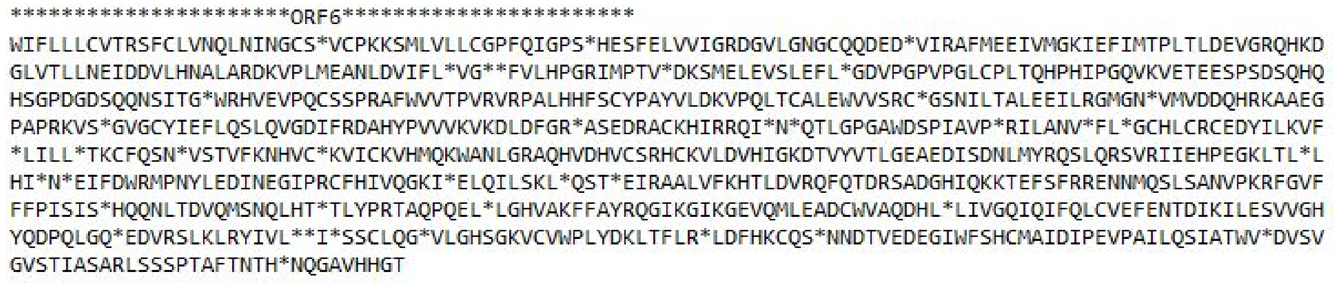
```
*********************ORF3**********************
FRDAQLLGFNECL*MQLVKNSGEQRQLWTPLQKRPIPMWQCSEE*QGPQECLWPYND*TKFPHLLLYHCFRTDTYENPII*GM*AYHKEAKRKPYQSAQGPTL
ANNYFRSITKQYI*ASDF*HPLIVQAEGPDNVLQQTPVS*YQCFQIQHRAGIFGFVPQ*AKGDLVPPNSQPQAFGPLL*CL*CPAYMQRIWQHVPTTVPGVER
FSGTKFKCAADCSFEHQ*GFAGVRRCLWGKRRPRISSAR*H*DSAYCFPVEKRIPFSSGCVRQHYDRFGTV*HQVCA*RPGLLLFLTCFVKAWKESEALKSYP
EQCGNNVEFLH*YPPDSLAYASQIFLNFKCEATRSTCLQDVQLF*HFSEGFVDTSSCH*CLQLPPKLHIQYLCQYEHPKLYNVWNTHGPHAVPVPS*PISACG
LYR*PFNRHGF*RL*KLS*IENT*FTKESVKKP*EYNPHICKDDIPTKTRH*PEFSKVQRWGNPMRLDPEFVSFKFVFEYAYRLCLQMLTSQSQGP*PSQQQD
NEHP*RCHPPAGFAGTQCSIQLLN*PSWVRGLQQPFCAGHRP*LSFPSL*GFLPELSEY*IPNSGKQPIPMHM*AEGLCQEHRLGSKRSGGGLA*LLQV*LPR
KL*GNCTEGLPHVSTIL*YCSADCHHRGHYAGAGCHWGFPLSLL*PALVCEDAVSVDTDQAQGQAHPLRGTPEKPPVPCFCLIQWA*FCLGEERITTQPRER*
HPDLPP*EELCPWQEHCGEHHQFH*EELQVHLCAVSPLHPE*VVSL*TLFCPSQSLP*RL**LNPHLAGTHSPVLHPYQLPQAQNSHVTKDLSGMAHREEQAW
TFLGKPKSIH*C*AG*PGRRNVLHTAIRIST
```



```
*********************ORF4**********************
GGYSYCCV*HVPSAWLTSLTLMDALRFAQKSPCLFFSVGHSR*VLRDMRVLSLW*LVGMEYWGMGASKMRIKLSEPSWKRL*WAK*SS**HHSLWMKWGDSTK
MDL*LFSMKLMMFSTMLLPGTKFLSWRQIWMSSFSRLGSNSFFTQAESCPLYETKAWNWRFLWSSSKGMCLALCLVCVH*HSILTYQGRSK*RQRKAPVTAST
SIVAPMVTVSRTVSQDSGDMWKSLSAVPLELSG*SHL*ESGQPSTTSLATQPMFLTKSLSSHVHWNGLFPAVRDLIF*QLWKKSSEGWETELWSMTSTERLLK
APHPGRSVKELDATLSSCKACRWVTSLGMLIILLL*RSRTLTLGGKHLKTEPVSIFEDKFKTNKLWVQAHGIPPSLYLREFWLMSSFCRDVIFADVRIIFSRF
FN*FFCKLSVFNLTKFLQSLKTMSVKRLSVKSTCRNGLTWDGHSMWTMCVPDIVKFWMFILAKILYM*LWGKLKTSVTT*CIDKAFREVSE*LNILKAS*PCS
FTFEIEKYLTGVCQTIWRILMKEFHVVSTLFRVRFESFRFFPSFDKARKK*EQPWSSSTHLMLDSSKPIVVLTDTSRRKRNSLFDGKTICRVSVLTCRRDSGS
SFSP*ASPNTSKTLLMFK*AISCTLELCTREPLNPRNCSWDMLPNSLHIGRASKALKERSKCLRLTVGWHKITFSSLWDKSKYSSSVLNLKTLILRYWSLL*D
IIRTLSLDNERMSEV*SSDILFCDRSKVVVCKGRSLGTLVRFAFGLFMISSHSSDNWIFISVSPKTMIQ*KMREFGLVIVWP*TFLRSLLFFRALPHGYRTFL
*GCPQLPLLA*VLHQLHSQTLIKTKELCITE
```



```
*********************ORF5**********************
VDILIAVCNTFLLPG*PA*H*WMLLGLPKKVHACSSLWAIPDRSFVT*EF*ACGNW*GWSTGEWVPAR*GLSYQSLHGRDCDGQNRVHNDTTHSG*SGETAQR
WTCNSSQ*N**CSPQCSCQGQSSSHGGKSGCHLSLGWVVIRSSPRQNHAHCMRQKHGTGGFSGVPLRGCAWPCAWSVSTDTASSHTRAGQSRDRGKPQ*QPAP
A*WPRW*QSAEQYHRIVETCGSPSVQFP*SFLGSHTCKSQASPPPLLLLPSLCS*QSPSAHMCIGMGCFPLLGI*YSDSSGRNPQRDGKLSYGR*PAQKGC*R
PRTQEGQLRSWMLH*VPAKPAGG*HL*GCSLSCCCEGQGP*LWEVSI*RQSL*AYSKTNLKLTNSGSRRMGFPHRCTLENSG*CLVFVGMSSLQM*GLYSQGF
LTDSFVN*VFSI*LSFYSL*KPCLLKGYL*SPHAEMG*LGTGTACGPCVFQTL*SFGCSYWQRYCICNFGGS*RHQ*QLDVSTKPSEKCQNN*TS*RQVDLVA
SHLKLRNI*LAYAKLSGGY**RNSTLFPHCSG*DLRASDSFQALTKHVRNKSSPGLQAHT*C*TVPNRS*C*RTHPEENGILFSTGKQYAESQC*RAEEIRGL
LFPHKHLLTPAKPY*CSNEQSAAHLNFVPENRSTPGTVVGTCCQILCI*AGHQRH*RRGPNA*G*LLGGTRSPLAHCGTNPNIPALC*I*KH*Y*DTGVCCRT
LSGPSAWTMRGCQKSEAQIYCFVIDLK*LFARVGPWALW*GLRLASL**AHIPQIIGFS*VSVLKQ*YSRR*GNLV*SLYGHRHS*GPCYSSEHCHMGIGRFC
RGVHNCLCSPEFFTNCIHKHSLKPRSCASRN
```



```
*********************ORF6**********************
WIFLLLCVTRSFCLVNQLNINGCS*VCPKKSMLVLLCGPFQIGPS*HESFELVVIGRDGVLGNGCQQDED*VIRAFMEEIVMGKIEFIMTPLTLDEVGRQHKD
GLVTLLNEIDDVLHNALARDKVPLMEANLDVIFL*VG**FVLHPGRIMPTV*DKSMELEVSLEFL*GDVPGPVPGLCPLTQHPHIPGQVKVETEESPSDSQHQ
HSGPDGDSQQNSITG*WRHVEVPQCSSPRAFWVVTPVRVRPALHHFSCYPAYVLDKVPQLTCALEWVVSRC*GSNILTALEEILRGMGN*VMVDDQHRKAAEG
PAPRKVS*GVGCYIEFLQSLQVGDIFRDAHYPVVVKVKDLDFGR*ASEDRACKHIRRQI*N*QTLGPGAWDSPIAVP*RILANV*FL*GCHLCRCEDYILKVF
*LILL*TKCFQSN*VSTVFKNHVC*KVICKVHMQKWANLGRAQHVDHVCSRHCKVLDVHIGKDTVYVTLGEAEDISDNLMYRQSLQRSVRIIEHPEGKLTL*L
HI*N*EIFDWRMPNYLEDINEGIPRCFHIVQGKI*ELQILSKL*QST*EIRAALVFKHTLDVRQFQTDRSADGHIQKKTEFSFRRENNMQSLSANVPKRFGVF
FFPISIS*HQQNLTDVQMSNQLHT*TLYPRTAQPQEL*LGHVAKFFAYRQGIKGIKGEVQMLEADCWVAQDHL*LIVGQIQIFQLCVEFENTDIKILESVVGH
YQDPQLGQ*EDVRSLKLRYIVL**I*SSCLQG*VLGHSGKVCVWPLYDKLTFLR*LDFHKCQS*NNDTVEDEGIWFSHCMAIDIPEVPAILQSIATWV*DVSV
GVSTIASARLSSSPTAFTNTH*NQGAVHHGT
```

Discussion

As you can see, the ORF1 consists of the longest protein sequence chain hence ORF1 is the acceptable reading frame corresponding to the actual protein sequence. I used the built in library function of seq from Bio.seq package for the translate() and reverse_complement() tasks performed on the sample gene sequence.

The results are counter checked from the *ExPasY Bioinformatics Resource Portal Tool*. https://web.expasy.org/translate/

Kanza Batool Haider
Independent Course Study
Supervisor: Prof. Mehmet Baysan
Date:2/7/2018

GLOBAL ALIGNMENT

The technique of Global Alignment is related to the Needleman–Wunsch algorithm, which is based on dynamic programming.

In this report and for my own understanding, I first manually performed the dynamic programming using two sample strings S1 and S2 on a scoring table and then used python programming to compute for Global Alignment all of its own, taking help from python's built in functions.

Manual Global Alignment and Score Table:

For the given two strings
S1= ACCGTT
S2= AGTTCA

	-	**A**	**C**	**C**	**G**	**T**	**T**
-	0	-1	-2	-3	-4	-5	-6
A	-1	1	0	-1	-2	-3	-4
G	-2	0	0	-1	0	-1	-2
T	-3	-1	-1	-1	-1	1	0
T	-4	-2	-2	-2	-2	0	2
C	-5	-3	-1	-1	-2	-1	1
A	-6	-4	-2	-2	-2	-2	0

```
ALIGNMENT:     ACCGTT_ _
               A_ _GTTCA
```

Technique:

The first position of the matrix [0,0] is by default = 0
The first row would '-1' itself and the score shall be added throughout the row
The first col would also '-1' itself and the score shall be added throughout the col
The interior scores are calculated based on the **max value** coming from either of the three:

Vertical: -1 from the vertical score
Horizontal: -1 from the horizontal score
Diagonal: -1 if the bases of s1 and s2 are a mismatch otherwise +1 in case of match

Kanza Batool Haider
Independent Course Study
Supervisor: Prof. Mehmet Baysan
Date:2/7/2018

Match (m)= 1
Mismatch (mm) = -1
Gap = -1

When the entire matrix has been scored, tracing back starts from the very last element of the row and col, tracing back from where the max value score was picked.

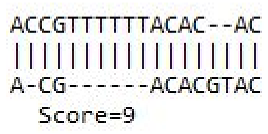
How to finally write the Alignment:
For a vertical direction of max score, give a gap in s1 (writing right to left)
For a horizontal direction of max score, give a gap in s2
For a diagonal direction of max score, this shows a match of the bases

```
ALIGNMENT:     ACCGTT_ _
               A_ _GTTCA
```

Implementation Code

```
from Bio import pairwise2
from Bio.pairwise2 import format_alignment
alignments = pairwise2.align.globalxx("ACCGTTTTTTACACAC", "ACGACACGTAC")
print(format_alignment(*alignments[0]))
```

Result



```
ACCGTTTTTTACAC--AC
||||||||||||||||||
A-CG------ACACGTAC
  Score=9
```

Discussion:

For the following sample of two strings there were total of 11 matches and a gap score of 9 as indicated in the result above.

Upcoming Week goal:

In the first week of the study, I learned and implemented the basic concepts of sequencing. The two main concepts worked upon were Protein sequencing and Global Alignment. Now that, I have the basic insight for sequencing a gene, in my upcoming week I shall be studying the format of a vcf file and how variants in the sample vcf files are compared with a reference file to discern useful information.