Course Code: CS 4115

Course Title: Computational Biology

Lab Sheet 01 - Report

Name : Nimna Alupotha Gamage (NIMNA A. G. T.)

Index No.: s14682

Reg. No. : 2019s17241

Degree : Bioinformatics

Year : 4th year

Lab sheet1: Information retrieval and sequence analysis

Q1

COX-2 (prostaglandin H2 synthase-2 (PTGS2)) gene

COX-2 has been thoroughly studied because of its role in prostaglandin synthesis. Prostaglandins have a wide range of roles in our body from aiding in digestion to propagating pain and inflammation.

Aspirin is a general inhibitor of prostaglandin synthesis and therefore, helps reduce pain.

However, aspirin also inhibits the synthesis of prostaglandins that aid in digestion. Therefore, aspirin is a poor choice for pain and inflammation management for those with ulcers or other digestion problems.

Recent advances in targeting specific prostaglandin-synthesizing enzymes have lead to the development of Celebrex, which is marketed as an arthritis therapy. Celebrex is a potent and specific inhibitor of COX-2. Celebrex is considered specific because it doesn’t inhibit COX-1, which is involved in synthesizing prostaglandins that aid in digestion.

This is a remarkable accomplishment given the great similarity between COX-1 and COX-2.

This achievement has paved the way for developing new therapies that bind more specifically to their target and therefore have fewer side effects. Understanding the enzyme structures of COX-1 and COX-2 helped researchers develop a drug that would only bind and inhibit COX-2. Many of the types of information and tools used by researchers for these types of studies are freely available on the web .

GenBank, SwissProt, Sequence Manipulation suite are some of the websites.

i. Access the entries for Human PTGS1 and PTGS2 in the “Gene” database at the NCBI (https://www.ncbi.nlm.nih.gov/) Website.

a. PTGS1 and PTGS2 are isozymes. Isozymes catalyze the same reaction but are separate genes. What types of reactions do PTGS enzymes catalyze? Also, what pathway are these enzymes a part of?

Types of reactions catalyzed by PTGS enzymes

PTGS (cyclooxygenase) is a key enzyme in prostaglandin biosynthesis, and it acts both as a dioxygenase and as a peroxidase. The isozymes PTGS1 and PTGS2, also known as COX-1 and COX-2, catalyze the conversion of arachidonic acid to prostaglandin H2 (PGH2). The insertion of two oxygen molecules into arachidonic acid results in the production of PGH2. This is an important step in the production of prostaglandins.

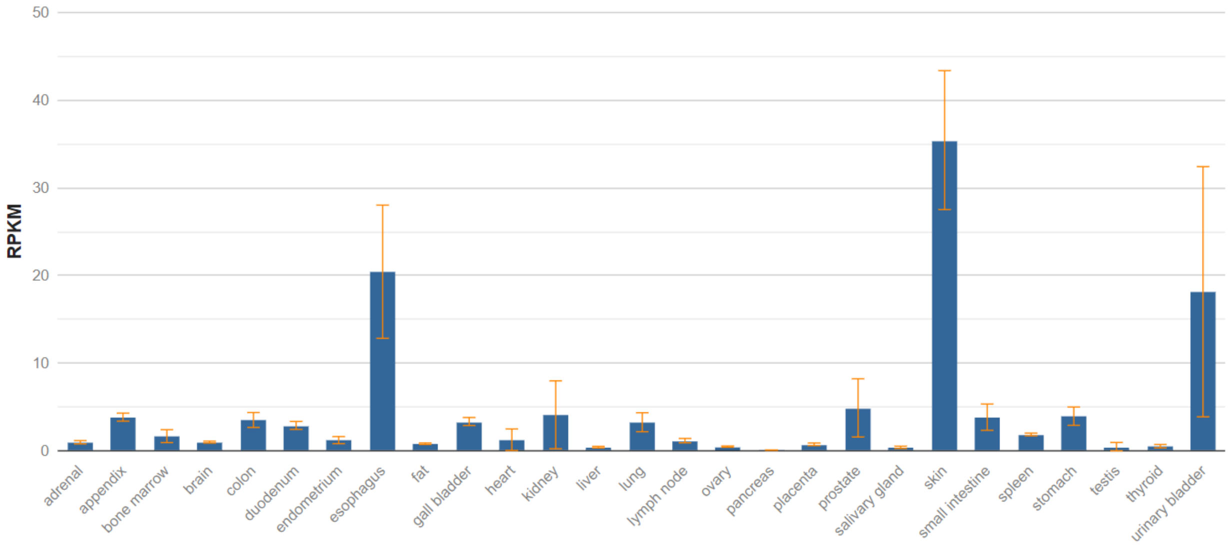
what pathway are these enzymes a part of?

These enzymes are a part of prostaglandin biosynthesis pathway. It is a is part of the wider eicosanoid biosynthesis pathway. Prostaglandins, thromboxanes, and leukotrienes are examples of eicosanoids, which are signalling molecules generated from arachidonic acid. Prostaglandins, in particular, have a wide range of physiological functions, including participation in inflammation, blood coagulation, blood pressure regulation, and other processes.

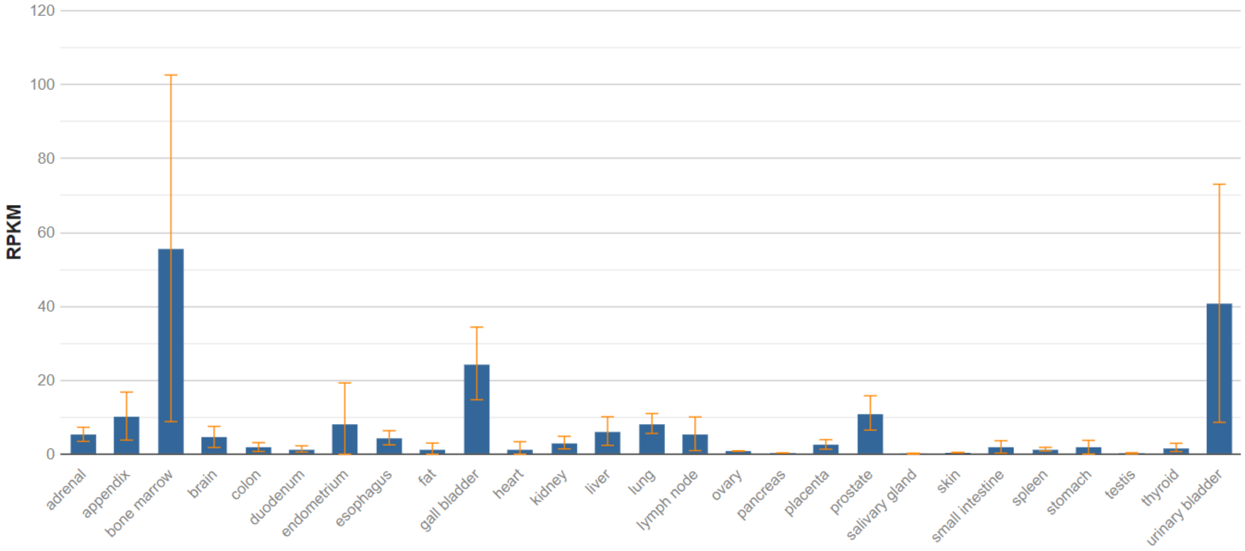
PTGS2 gene encodes the inducible isozyme. It is regulated by specific stimulatory events, suggesting that it is responsible for the prostanoid biosynthesis involved in inflammation and mitogenesis.

b. How is the expression of PTGS1 and PTGS2 different?

PTGS1 is biased expression in skin, esophagus and 12 other tissues.



But PTGS2 is biased expression in bone marrow, urinary bladder and 11 other tissues.



c. Which isozyme (PTGS1 or PTGS2) is required to inhibit inflammation?

PTGS2

d. The drug Celebrex selectively inhibits PTGS2 while aspirin and other NSAID's inhibit both PTGS1 and PTGS2 in the same way. Why do you think researchers wanted to discover a selective inhibitor to PTGS2?

PTGS1 is constitutively expressed in numerous tissues and is engaged in basic physiological activities such as stomach lining protection and blood clotting regulation. Researchers tried to **lessen the negative effects** associated with non-selective inhibition of PTGS2 by designing selective inhibitors.

PTGS1 performs housekeeping activities in a variety of tissues, and decreasing its activity may disturb normal physiological processes. By selectively targeting PTGS2, researchers hoped to control the inflammatory response without interfering with PTGS1's positive effects in homeostasis maintenance. This selectivity was predicted to give a more tailored and safer approach to inflammation management **without interfering with key physiological functions**.

e. Describe how studying 3-D structures of PTGS1 and PTGS2 could help researchers design a drug that binds to PTGS1, but not to PTGS2.

An enzyme's active site is the region where substrate/ drug binding and catalysis take place. Researchers can find **differences in the active sites** of PTGS1 and PTGS2 by analysing the 3D structures of the two isozymes.

Understanding the structural differences between the active sites of PTGS1 and PTGS2 allows researchers to create compounds that preferentially interact with one isozyme while avoiding or minimising interactions with the other.

The 3D structures can show the form, size, and chemical characteristics of the PTGS1 and PTGS2 binding sites. Differences in the **binding site architecture** of the two isozymes can be used to create drugs that fit preferentially into the PTGS1 active site but not the PTGS2 active site.

ii. Considering the Homo sapiens PTGS2 gene entry in NCBI gene https://www.ncbi.nlm.nih.gov/gene/ database,

a. What is the gene name?

Official Full Name: prostaglandin-endoperoxide synthase 2

b. What is the GeneID number?

Gene ID: 5743

c. Where in the human genome is this gene located?

Location: 1q31.1 [Chromosome 1 - NC_000001.11]

d. What is the RefSeq accession number for the mRNA sequence of H o m o sa pie n s prostaglandin-endoperoxide synthase 2?

NM_000963.4

e. Download the prostaglandin-endoperoxide synthase 2 Reference mRNA sequence in “FASTA” format.

```
>NM_000963.4 Homo sapiens prostaglandin-endoperoxide synthase 2 (PTGS2), mRNA
AATTGTCATACGACTTGCAGTGAGCGTCAGGAGCACGTCCAGGAACTCCTCAGCAGCGCCTCCTTCAGCT
CCACAGCCAGACGCCCTCAGACAGCAAAGCCTACCCCCGCGCCGCGCCCTGCCCGCCGCTGCGATGCTCG
CCCGCGCCCTGCTGCTGTGCGCGGTCCTGGCGCTCAGCCATACAGCAAATCCTTGCTGTTCCCACCCATG
TCAAAACCGAGGTGTATGTATGAGTGTGGGATTTGACCAGTATAAGTGCGATTGTACCCGGACAGGATTC
TATGGAGAAAACTGCTCAACACCGGAATTTTTGACAAGAATAAAATTATTTCTGAAACCCACTCCAAACA
CAGTGCACTACATACTTACCCACTTCAAGGGATTTTGGAACGTTGTGAATAACATTCCCTTCCTTCGAAA
TGCAATTATGAGTTATGTGTTGACATCCAGATCACATTTGATTGACAGTCCACCAACTTACAATGCTGAC
TATGGCTACAAAAGCTGGGAAGCCTTCTCTAACCTCTCCTATTATACTAGAGCCCTTCCTCCTGTGCCTG
ATGATTGCCCGACTCCCTTGGGTGTCAAAGGTAAAAAGCAGCTTCCTGATTCAAATGAGATTGTGGAAAA
ATTGCTTCTAAGAAGAAAGTTCATCCCTGATCCCCAGGGCTCAAACATGATGTTTGCATTCTTTGCCCAG
CACTTCACGCATCAGTTTTTCAAGACAGATCATAAGCGAGGGCCAGCTTTCACCAACGGGCTGGGCCATG
GGGTGGACTTAAATCATATTTACGGTGAAACTCTGGCTAGACAGCGTAAACTGCGCCTTTTCAAGGATGG
AAAAATGAAATATCAGATAATTGATGGAGAGATGTATCCTCCCACAGTCAAAGATACTCAGGCAGAGATG
ATCTACCCTCCTCAAGTCCCTGAGCATCTACGGTTTGCTGTGGGGCAGGAGGTCTTTGGTCTGGTGCCTG
GTCTGATGATGTATGCCACAATCTGGCTGCGGGAACACAACAGAGTATGCGATGTGCTTAAACAGGAGCA
TCCTGAATGGGGTGATGAGCAGTTGTTCCAGACAAGCAGGCTAATACTGATAGGAGAGACTATTAAGATT
GTGATTGAAGATTATGTGCAACACTTGAGTGGCTATCACTTCAAACTGAAATTTGACCCAGAACTACTTT
TCAACAAACAATTCCAGTACCAAAATCGTATTGCTGCTGAATTTAACACCCTCTATCACTGGCATCCCCT
TCTGCCTGACACCTTTCAAATTCATGACCAGAAATACAACTATCAACAGTTTATCTACAACAACTCTATA
TTGCTGGAACATGGAATTACCCAGTTTGTTGAATCATTCACCAGGCAAATTGCTGGCAGGGTTGCTGGTG
GTAGGAATGTTCCACCCGCAGTACAGAAAGTATCACAGGCTTCCATTGACCAGAGCAGGCAGATGAAATA
CCAGTCTTTTAATGAGTACCGCAAACGCTTTATGCTGAAGCCCTATGAATCATTTGAAGAACTTACAGGA
GAAAAGGAAATGTCTGCAGAGTTGGAAGCACTCTATGGTGACATCGATGCTGTGGAGCTGTATCCTGCCC
TTCTGGTAGAAAAGCCTCGGCCAGATGCCATCTTTGGTGAAACCATGGTAGAAGTTGGAGCACCATTCTC
CTTGAAAGGACTTATGGGTAATGTTATATGTTCTCCTGCCTACTGGAAGCCAAGCACTTTTGGTGGAGAA
GTGGGTTTTCAAATCATCAACACTGCCTCAATTCAGTCTCTCATCTGCAATAACGTGAAGGGCTGTCCCT
TTACTTCATTCAGTGTTCCAGATCCAGAGCTCATTAAAACAGTCACCATCAATGCAAGTTCTTCCCGCTC
CGGACTAGATGATATCAATCCCACAGTACTACTAAAAGAACGTTCGACTGAACTGTAGAAGTCTAATGAT
CATATTTATTTATTTATATGAACCATGTCTATTAATTTAATTATTTAATAATATTTATATTAAACTCCTT
ATGTTACTTAACATCTTCTGTAACAGAAGTCAGTACTCCTGTTGCGGAGAAAGGAGTCATACTTGTGAAG
ACTTTTATGTCACTACTCTAAAGATTTTGCTGTTGCTGTTAAGTTTGGAAAACAGTTTTTATTCTGTTTT
ATAAACCAGAGAGAAATGAGTTTTGACGTCTTTTTACTTGAATTTCAACTTATATTATAAGAACGAAAGT
AAAGATGTTTGAATACTTAAACACTGTCACAAGATGGCAAAATGCTGAAAGTTTTTACACTGTCGATGTT
TCCAATGCATCTTCCATGATGCATTAGAAGTAACTAATGTTTGAAATTTTAAAGTACTTTTGGTTATTTT
TCTGTCATCAAACAAAAACAGGTATCAGTGCATTATTAAATGAATATTTAAATTAGACATTACCAGTAAT
TTCATGTCTACTTTTTAAAATCAGCAATGAAACAATAATTTGAAATTTCTAAATTCATAGGGTAGAATCA
CCTGTAAAAGCTTGTTTGATTTCTTAAAGTTATTAAACTTGTACATATACCAAAAAGAAGCTGTCTTGGA
TTTAAATCTGTAAAATCAGTAGAAATTTTACTACAATTGCTTGTTAAAATATTTTATAAGTGATGTTCCT
TTTTCACCAAGAGTATAAACCTTTTTAGTGTGACTGTTAAAACTTCCTTTTAAATCAAAATGCCAAATTT
ATTAAGGTGGTGGAGCCACTGCAGTGTTATCTTAAAATAAGAATATTTTGTTGAGATATTCCAGAATTTG
TTTATATGGCTGGTAACATGTAAAATCTATATCAGCAAAAGGGTCTACCTTTAAAATAAGCAATAACAAA
GAAGAAAACCAAATTATTGTTCAAATTTAGGTTTAAACTTTTGAAGCAAACTTTTTTTTATCCTTGTGCA
CTGCAGGCCTGGTACTCAGATTTTGCTATGAGGTTAATGAAGTACCAAGCTGTGCTTGAATAATGATATG
TTTTCTCAGATTTTCTGTTGTACAGTTTAATTTAGCAGTCCATATCACATTGCAAAAGTAGCAATGACCT
CATAAAATACCTCTTCAAAATGCTTAAATTCATTTCACACATTAATTTTATCTCAGTCTTGAAGCCAATT
CAGTAGGTGCATTGGAATCAAGCCTGGCTACCTGCATGCTGTTCCTTTTCTTTTCTTCTTTTAGCCATTT
```

```
TGCTAAGAGACACAGTCTTCTCATCACTTCGTTTCTCCTATTTTGTTTTACTAGTTTTAAGATCAGAGTT
CACTTTCTTTGGACTCTGCCTATATTTTCTTACCTGAACTTTTGCAAGTTTTCAGGTAAACCTCAGCTCA
GGACTGCTATTTAGCTCCTCTTAAGAAGATTAAAAGAGAAAAAAAAAGGCCCTTTTAAAAATAGTATACA
CTTATTTTAAGTGAAAAGCAGAGAATTTTATTTATAGCTAATTTTAGCTATCTGTAACCAAGATGGATGC
AAAGAGGCTAGTGCCTCAGAGAGAACTGTACGGGGTTTGTGACTGGAAAAAGTTACGTTCCCATTCTAAT
TAATGCCCTTTCTTATTTAAAAACAAAACCAAATGATATCTAAGTAGTTCTCAGCAATAATAATAATGAC
GATAATACTTCTTTTCCACATCTCATTGTCACTGACATTTAATGGTACTGTATATTACTTAATTTATTGA
AGATTATTATTTATGTCTTATTAGGACACTATGGTTATAAACTGTGTTTAAGCCTACAATCATTGATTTT
TTTTTGTTATGTCACAATCAGTATATTTTCTTTGGGGTTACCTCTCTGAATATTATGTAAACAATCCAAA
GAAATGATTGTATTAAGATTTGTGAATAAATTTTTAGAAATCTGATTGGCATATTGAGATATTTAAGGTT
GAATGTTTGTCCTTAGGATAGGCCTATGTGCTAGCCCACAAAGAATATTGTCTCATTAGCCTGAATGTGC
CATAAGACTGACCTTTTAAAATGTTTTGAGGGATCTGTGGATGCTTCGTTAATTTGTTCAGCCACAATTT
ATTGAGAAAATATTCTGTGTCAAGCACTGTGGGTTTTAATATTTTTAAATCAAACGCTGATTACAGATAA
TAGTATTTATATAAATAATTGAAAAAAATTTTCTTTTGGGAAGAGGGAGAAAATGAAATAAATATCATTA
AAGATAACTCAGGAGAATCTTCTTTACAATTTTACGTTTAGAATGTTTAAGGTTAAGAAAGAAATAGTCA
ATATGCTTGTATAAAACACTGTTCACTGTTTTTTTTAAAAAAAAAACTTGATTTGTTATTAACATTGATC
TGCTGACAAAACCTGGGAATTTGGGTTGTGTATGCGAATGTTTCAGTGCCTCAGACAAATGTGTATTTAA
CTTATGTAAAAGATAAGTCTGGAAATAAATGTCTGTTTATTTTTGTACTATTTAAAAATTGACAGATCTT
TTCTGAAGATAAACTTTGATTGTTTCTATA
```

Downloaded fasta file name: “sequence_PTGS2_mRNA.fasta”

f. What is the RefSeq accession number for the Homo sapiens PTGS2 protein sequence? Download the sequence in “FASTA” format.

RefSeq accession number: NP_000954.1

Sequence in “FASTA” format:

```
>NP_000954.1 prostaglandin G/H synthase 2 precursor [Homo sapiens]
MLARALLLCAVLALSHTANPCCSHPCQNRGVCMSVGFDQYKCDCTRTGFYGENCSTPEFLTRIKLFLKPT
PNTVHYILTHFKGFWNVVNNIPFLRNAIMSYVLTSRSHLIDSPPTYNADYGYKSWEAFSNLSYYTRALPP
VPDDCPTPLGVKGKKQLPDSNEIVEKLLLRRKFIPDPQGSNMMFAFFAQHFTHQFFKTDHKRGPAFTNGL
GHGVDLNHIYGETLARQRKLRLFKDGKMKYQIIDGEMYPPTVKDTQAEMIYPPQVPEHLRFAVGQEVFGL
VPGLMMYATIWLREHNRVCDVLKQEHPEWGDEQLFQTSRLILIGETIKIVIEDYVQHLSGYHFKLKFDPE
LLFNKQFQYQNRIAAEFNTLYHWHPLLPDTFQIHDQKYNYQQFIYNNSILLEHGITQFVESFTRQIAGRV
AGGRNVPPAVQKVSQASIDQSRQMKYQSFNEYRKRFMLKPYESFEELTGEKEMSAELEALYGDIDAVELY
PALLVEKPRPDAIFGETMVEVGAPFSLKGLMGNVICSPAYWKPSTFGGEVGFQIINTASIQSLICNNVKG
CPFTSFSVPDPELIKTVTINASSSRSGLDDINPTVLLKERSTEL
```

Downloaded fasta file name: “sequence_PTGS2_protein.fasta”

iii. Search for the UniProt entry for PTGS2 in Expasy https://www.expasy.org/website.

a. What are the alternate names for this protein.

Prostaglandin G/H synthase 2, 1.14.99.1, Cyclooxygenase-2, COX-2, PHS II, Prostaglandin H2 synthase 2, PGH synthase 2, PGHS-2, Prostaglandin-endoperoxide synthase 2

b. What types of drugs target this protein?

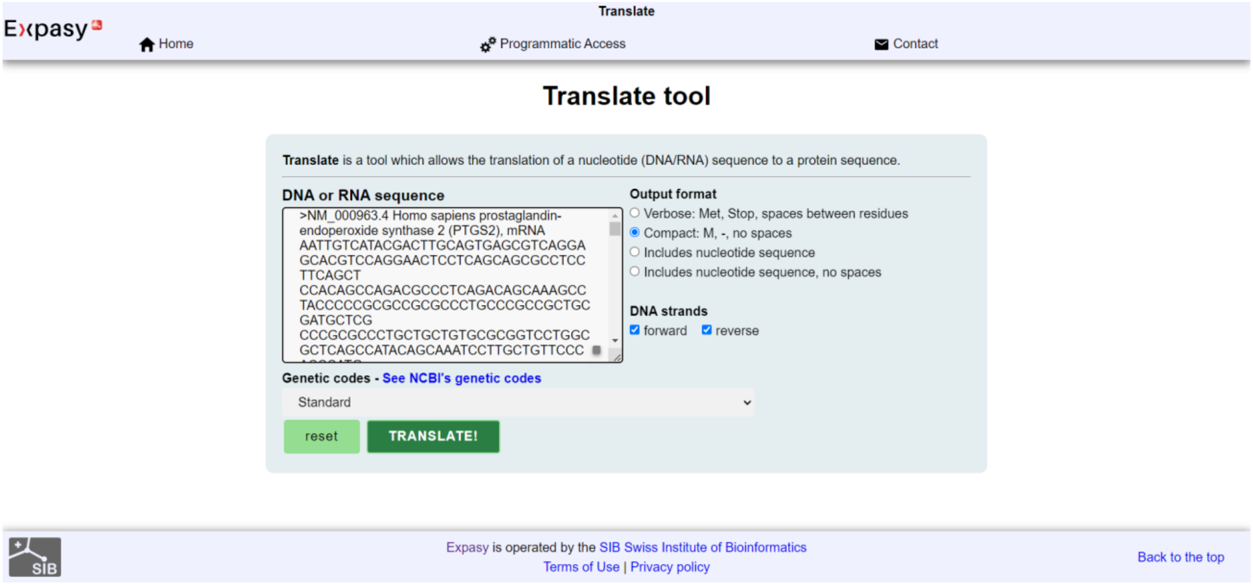
Nonsteroidal anti-inflammatory drugs (NSAIDs) including aspirin and ibuprofen, are types of drugs target PTGS1 and PTGS2 proteins

c. What amino acid is acetylated by aspirin (amino acid type)?

Serine amino acid is acetylated by aspirin. Aspirin is able to produce an irreversible inactivation of the enzyme through a serine acetylation

iv. Translate the mRNA sequence of PTGS2 into Protein. Use "Translate " tool in ExPASy. Explain the output.

Input mRNA sequence and parameters;


Expasy
Translate
Home
Programmatic Access
Contact
reset
TRANSLATE!
Expasy is operated by the SIB Swiss Institute of Bioinformatics
Terms of Use | Privacy policy
Back to the top

Translate tool

Translate is a tool which allows the translation of a nucleotide (DNA/RNA) sequence to a protein sequence.

DNA or RNA sequence

```
>NM_000963.4 Homo sapiens prostaglandin-
endoperoxide synthase 2 (PTGS2), mRNA
AATTGTCATACGACTTGCAGTGAGCGTCAGGA
GCACGTCCAGGAACTCCTCAGCAGCGCCTCC
TTCAGCT
CCACAGCCAGACGCCCTCAGACAGCAAAGCC
TACCCCCGCGCCGCGCCCTGCCCGCCGCTGC
GATGCTCG
CCCGCGCCCTGCTGCTGTGCGCGGTCCTGGC
GCTCAGCCATACAGCAAATCCTTGCTGTTCCC
```

Output format

○ Verbose: Met, Stop, spaces between residues
● Compact: M, -, no spaces
○ Includes nucleotide sequence
○ Includes nucleotide sequence, no spaces

DNA strands

☑ forward ☑ reverse

Genetic codes - See NCBI's genetic codes

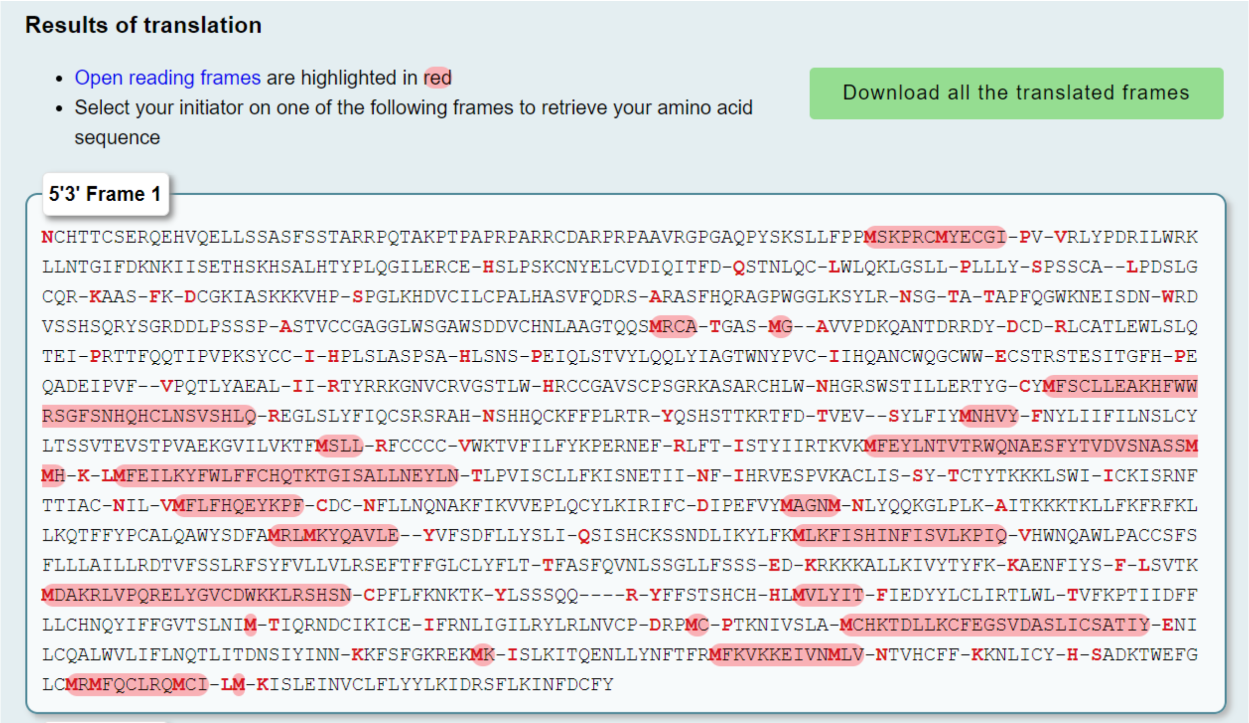
Standard

Results of translation;

The input mRNA sequence is translated in **all six open reading frames**. Three frames of 5'3' direction and the other three frames for 3'5' direction.

The correct reading frame is the **longest continuous reading frame**.

The longest continuous reading frame of this output belongs to **5'3' Frame 2.** It is surrounded by a blue color rectangle.

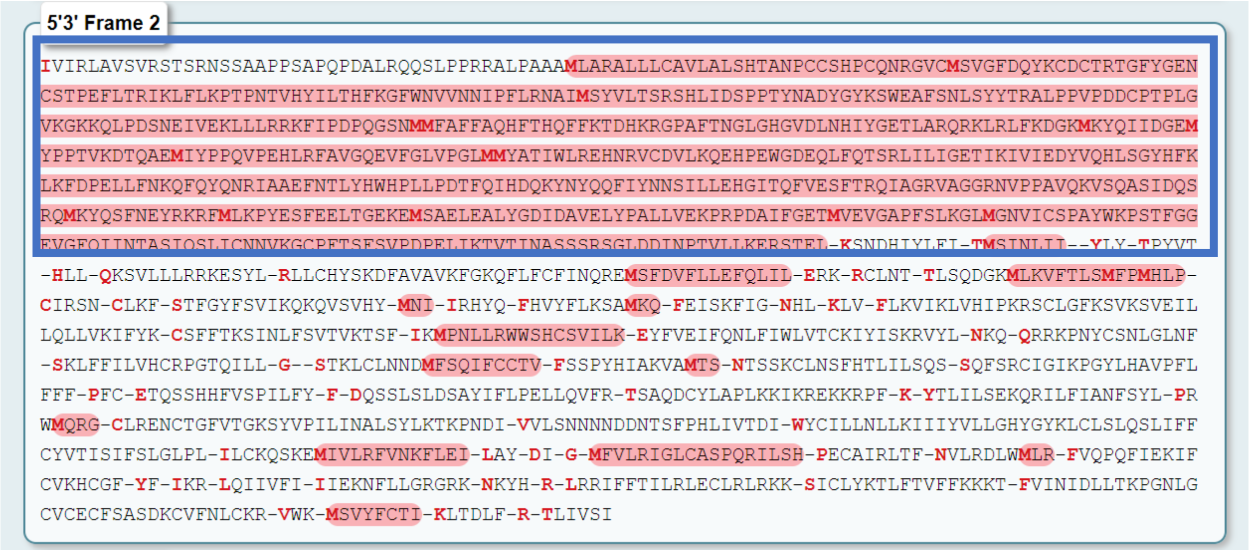


Results of translation

- Open reading frames are highlighted in red
- Select your initiator on one of the following frames to retrieve your amino acid sequence

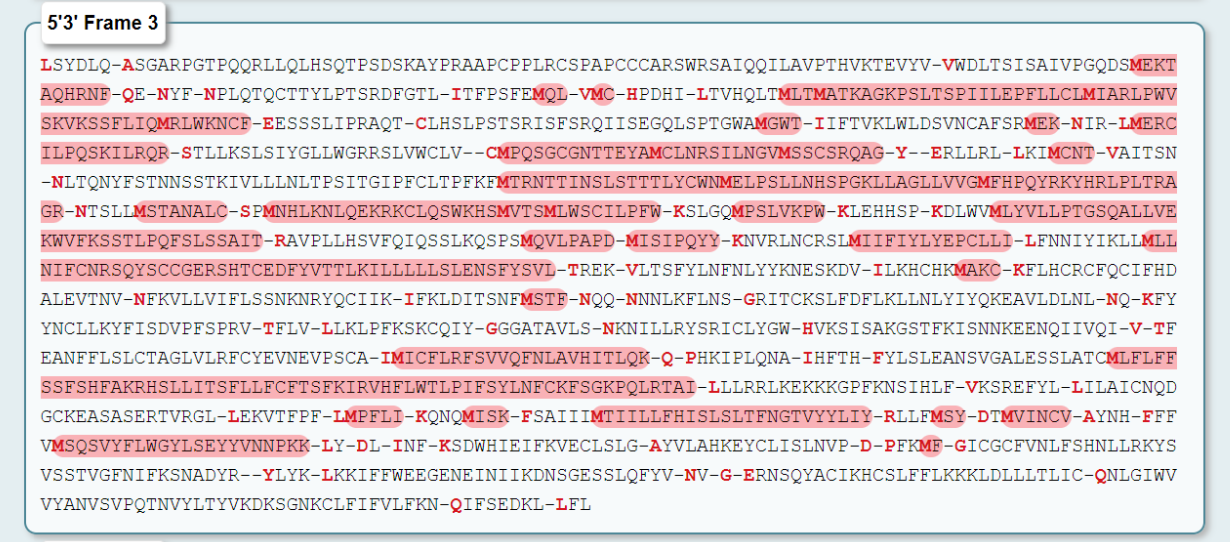
5'3' Frame 1

```
NCHTTCSERQEHVQELLSSASFSSTARRPQTAKPTPAPRPARRCDARPRPAAVRGPGAQPYSKSLLFPPMSKPRCMYECGI-PV-VRLYPDRILWRK
LLNTGIFDKNKIISETHSKHSALHTYPLQGILERCE-HSLPSKCNYELCVDIQITFD-QSTNLQC-LWLQKLGSLL-PLLLY-SPSSCA--LPDSLG
CQR-KAAS-FK-DCGKIASKKKVHP-SPGLKHDVCILCPALHASVFQDRS-ARASFHQRAGPWGGLKSYLR-NSG-TA-TAPFQGWKNEISDN-WRD
VSSHSQRYSGRDDLPSSSP-ASTVCCGAGGLWSGAWSDDVCHNLAAGTQQSMRCA-TGAS-MG--AVVPDKQANTDRRDY-DCD-RLCATLEWLSLQ
TEI-PRTTFQQTIPVPKSYCC-I-HPLSLASPSA-HLSNS-PEIQLSTVYLQQLYIAGTWNYPVC-IIHQANCWQGCWW-ECSTRSTESITGFH-PE
QADEIPVF--VPQTLYAEAL-II-RTYRRKGNVCRVGSTLW-HRCCGAVSCPSGRKASARCHLW-NHGRSWSTILLERTYG-CYMFSCLLEAKHFWW
RSGFSNHQHCLNSVSHLQ-REGLSLYFIQCSRSRAH-NSHHQCKFFPLRTR-YQSHSTTKRTFD-TVEV--SYLFIYMNHVY-FNYLIIFILNSLCY
LTSSVTEVSTPVAEKGVILVKTFMSLL-RFCCCC-VWKTVFILFYKPERNEF-RLFT-ISTYIIRTKVKMFEYLNTVTRWQNAESFYTVDVSNASSM
MH-K-LMFEILKYFWLFFCHQTKTGISALLNEYLN-TLPVISCLLFKISNETII-NF-IHRVESPVKACLIS-SY-TCTYTKKKLSWI-ICKISRNF
TTIAC-NIL-VMFLFHQEYKPF-CDC-NFLLNQNAKFIKVVEPLQCYLKIRIFC-DIPEFVYMAGNM-NLYQQKGLPLK-AITKKKTKLLFKFRFKL
LKQTFFYPCALQAWYSDFAMRLMKYQAVLE--YVFSDFLLYSLI-QSISHCKSSNDLIKYLFKMLKFISHINFISVLKPIQ-VHWNQAWLPACCSFS
FLLLAILLRDTVFSSLRFSYFVLLVLRSEFTFFGLCLYFLT-TFASFQVNLSSGLLFSSS-ED-KRKKKALLKIVYTYFK-KAENFIYS-F-LSVTK
MDAKRLVPQRELYGVCDWKKLRSHSN-CPFLFKNKTK-YLSSSQQ----R-YFFSTSHCH-HLMVLYIT-FIEDYYLCLIRTLWL-TVFKPTIIDFF
LLCHNQYIFFGVTSLNIM-TIQRNDCIKICE-IFRNLIGILRYLRLNVCP-DRPMC-PTKNIVSLA-MCHKTDLLKCFEGSVDASLICSATIY-ENI
LCQALWVLIFLNQTLITDNSIYINN-KKFSFGKREKMK-ISLKITQENLLYNFTFRMFKVKKEIVNMLV-NTVHCFF-KKNLICY-H-SADKTWEFG
LCMRMFQCLRQMCI-LM-KISLEINVCLFLYYLKIDRSFLKINFDCFY
```



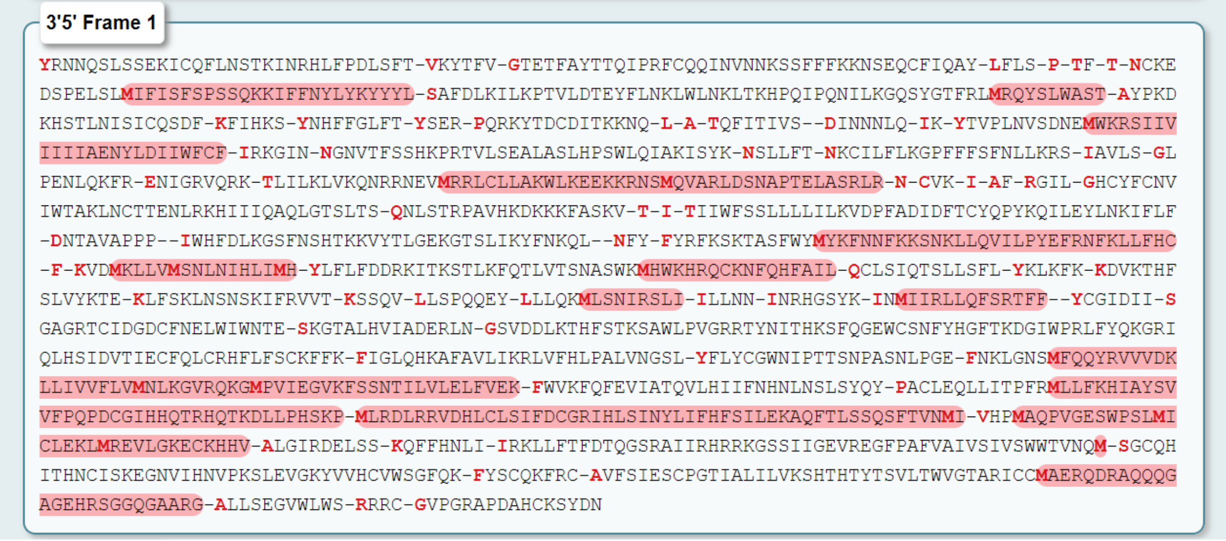
5'3' Frame 2

```
IVIRLAVSVRSTSRNSSAAPPSAPQPDALRQQSLPPRRALPAAAMLARALLLCAVLALSHTANPCCSHPCQNRGVCMSVGFDQYKCDCTRTGFYGEN
CSTPEFLTRIKLFLKPTPNTVHYILTHFKGFWNVVNNIPFLRNAIMSYVLTSRSHLIDSPPTYNADYGYKSWEAFSNLSYYTRALPPVPDDCPTPLG
VKGKKQLPDSNEIVEKLLLRRKFIPDPQGSNMMFAFFAQHFTHQFFKTDHKRGPAFTNGLGHGVDLNHIYGETLARQRKLRLFKDGKMKYQIIDGEM
YPPTVKDTQAEMIYPPQVPEHLRFAVGQEVFGLVPGLMMYATIWLREHNRVCDVLKQEHPEWGDEQLFQTSRLILIGETIKIVIEDYVQHLSGYHFK
LKFDPELLFNKQFQYQNRIAAEFNTLYHWHPLLPDTFQIHDQKYNYQQFIYNNSILLEHGITQFVESFTRQIAGRVAGGRNVPPAVQKVSQASIDQS
RQMKYQSFNEYRKRFMLKPYESFEELTGEKEMSAELEALYGDIDAVELYPALLVEKPRPDAIFGETMVEVGAPFSLKGLMGNVICSPAYWKPSTFGG
EVGFQLINTASIQSLICNNVKGCPFTSFSVPDPELIKTVTINASSSRSGLDDINPTVLLKERSTEL-KSNDHIYLFI-TMSINLII--YLY-TPYVT
-HLL-QKSVLLLRRKESYL-RLLCHYSKDFAVAVKFGKQFLFCFINQREMSFDVFLLEFQLIL-ERK-RCLNT-TLSQDGKMLKVFTLSMFPMHLP-
CIRSN-CLKF-STFGYFSVIKQKQVSVHY-MNI-IRHYQ-FHVYFLKSAMKQ-FEISKFIG-NHL-KLV-FLKVIKLVHIPKRSCLGFKSVKSVEIL
LQLLVKIFYK-CSFFTKSINLFSVTVKTSF-IKMPNLLRWWSHCSVILK-EYFVEIFQNLFIWLVTCKIYISKRVYL-NKQ-QRRKPNYCSNLGLNF
-SKLFFILVHCRPGTQILL-G--STKLCLNNDMFSQIFCCTV-FSSPYHIAKVAMTS-NTSSKCLNSFHTLILSQS-SQFSRCIGIKPGYLHAVPFL
FFF-PFC-ETQSSHHFVSPILFY-F-DQSSLSLDSAYIFLPELLQVFR-TSAQDCYLAPLKKIKREKKRPF-K-YTLILSEKQRILFIANFSYL-PR
WMQRG-CLRENCTGFVTGKSYVPILINALSYLKTKPNDI-VVLSNNNNDDNTSFPHLIVTDI-WYCILLNLLKIIIYVLLGHYGYKLCLSLQSLIFF
CYVTISIFSLGLPL-ILCKQSKEMIVLRFVNKFLEI-LAY-DI-G-MFVLRIGLCASPQRILSH-PECAIRLTF-NVLRDLWMLR-FVQPQFIEKIF
CVKHCGF-YF-IKR-LQIIVFI-IIEKNFLLGRGRK-NKYH-R-LRRIFFTILRLECLRLRKK-SICLYKTLFTVFFKKKT-FVINIDLLTKPGNLG
CVCECFSASDKCVFNLCKR-VWK-MSVYFCTI-KLTDLF-R-TLIVSI
```



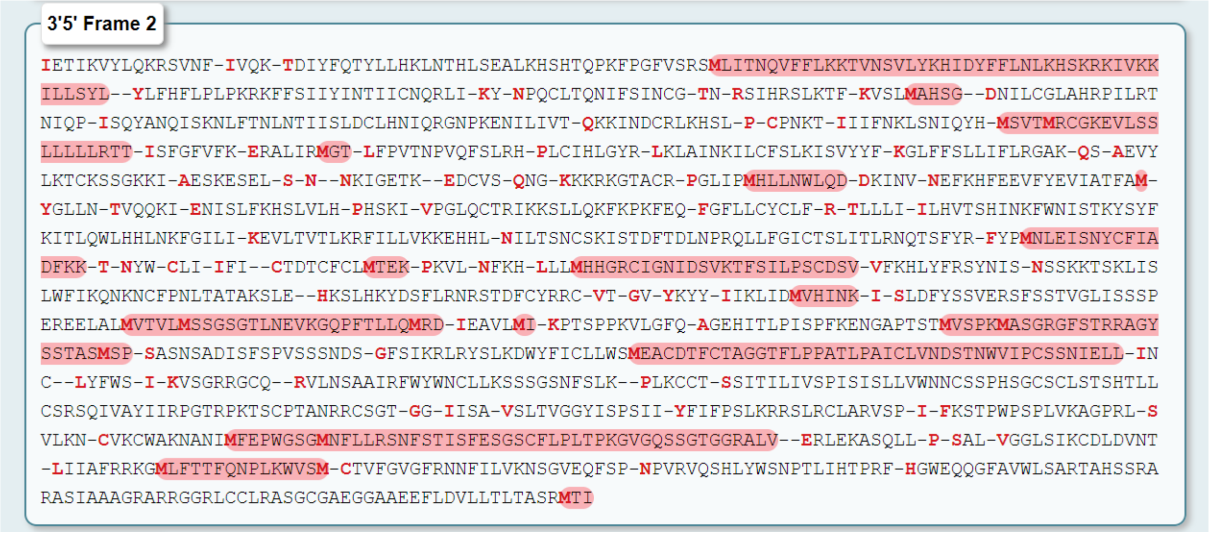
5'3' Frame 3

```
LSYDLQ-ASGARPGTPQQRLLQLHSQTPSDSKAYPRAAPCPPLRCSPAPCCCARSWRSAIQQILAVPTHVKTEVYV-VWDLTSISAIVPGQDSMEKT
AQHRNF-QE-NYF-NPLQTQCTTYLPTSRDFGTL-ITFPSFEMQL-VMC-HPDHI-LTVHQLTMLTMATKAGKPSLTSPIILEPFLLCLMIARLPWV
SKVKSSFLIQMRLWKNCF-EESSSLIPRAQT-CLHSLPSTSRISFSRQIISEGQLSPTGWAMGWT-IIFTVKLWLDSVNCAFSRMEK-NIR-LMERC
ILPQSKILRQR-STLLKSLSIYGLLWGRRSLVWCLV--CMPQSGCGNTTEYAMCLNRSILNGVMSSCSRQAG-Y--ERLLRL-LKIMCNT-VAITSN
-NLTQNYFSTNNSSTKIVLLLNLTPSITGIPFCLTPFKFMTRNTTINSLSTTTLYCWNMELPSLLNHSPGKLLAGLLVVGMFHPQYRKYHRLPLTRA
GR-NTSLLMSTANALC-SPMNHLKNLQEKRKCLQSWKHSMVTSMLWSCILPFW-KSLGQMPSLVKPW-KLEHHSP-KDLWVMLYVLLPTGSQALLVE
KWVFKSSTLPQFSLSSAIT-RAVPLLHSVFQIQSSLKQSPSMQVLPAPD-MISIPQYY-KNVRLNCRSLMIIFIYLYEPCLLI-LFNNIYIKLLMLL
NIFCNRSQYSCCGERSHTCEDFYVTTLKILLLLLSLENSFYSVL-TREK-VLTSFYLNFNLYYKNESKDV-ILKHCHKMAKC-KFLHCRCFQCIFHD
ALEVTNV-NFKVLLVIFLSSNKNRYQCIIK-IFKLDITSNFMSTF-NQQ-NNNLKFLNS-GRITCKSLFDFLKLLNLYIYQKEAVLDLNL-NQ-KFY
YNCLLKYFISDVPFSPRV-TFLV-LLKLPFKSKCQIY-GGGATAVLS-NKNILLRYSRICLYGW-HVKSISAKGSTFKISNNKEENQIIVQI-V-TF
EANFFLSLCTAGLVLRFCYEVNEVPSCA-IMICFLRFSVVQFNLAVHITLQK-Q-PHKIPLQNA-IHFTH-FYLSLEANSVGALESSLATCMLFLFF
SSFSHFAKRHSLLITSFLLFCFTSFKIRVHFLWTLPIFSYLNFCKFSGKPQLRTAI-LLLRRLKEKKKGPFKNSIHLF-VKSREFYL-LILAICNQD
GCKEASASERTVRGL-LEKVTFPF-LMPFLI-KQNQMISK-FSAIIIMTIILLFHISLSLTFNGTVYYLIY-RLLFMSY-DTMVINCV-AYNH-FFF
VMSQSVYFLWGYLSEYYVNNPKK-LY-DL-INF-KSDWHIEIFKVECLSLG-AYVLAHKEYCLISLNVP-D-PFKMF-GICGCFVNLFSHNLLRKYS
VSSTVGFNIFKSNADYR--YLYK-LKKIFFWEEGENEINIIKDNSGESSLQFYV-NV-G-ERNSQYACIKHCSLFFLKKKLDLLLTLIC-QNLGIWV
VYANVSVPQTNVYLTYVKDKSGNKCLFIFVLFKN-QIFSEDKL-LFL
```



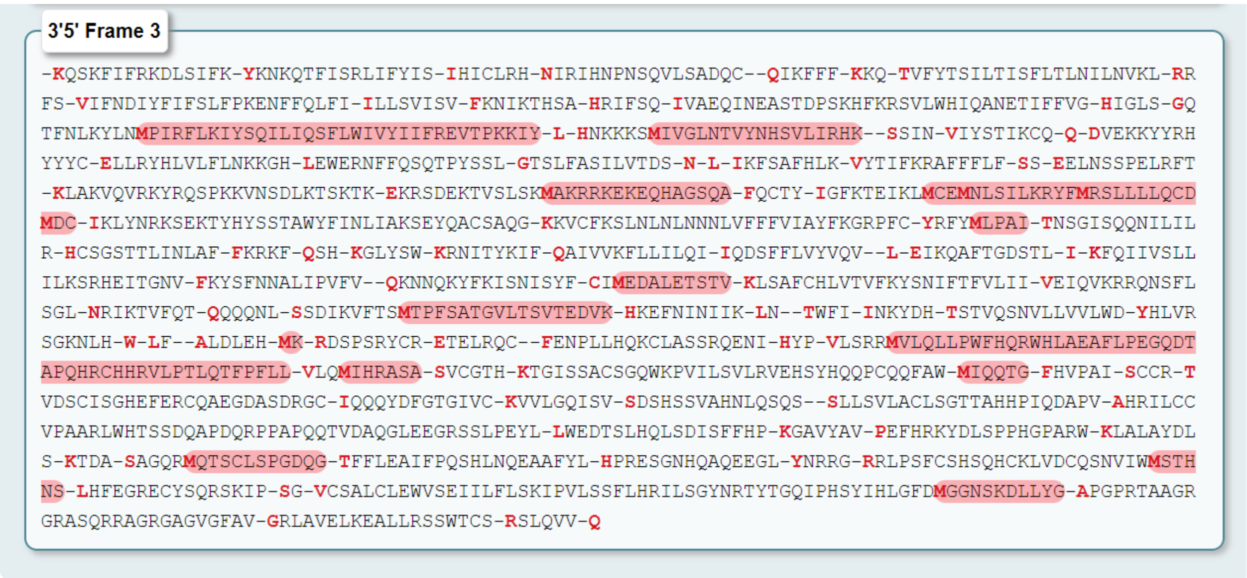
3'5' Frame 1

```
YRNNQSLSSEKICQFLNSTKINRHLFPDLSFT-VKYTFV-GTETFAYTTQIPRFCQQINVNNKSSFFFKKNSEQCFIQAY-LFLS-P-TF-T-NCKE
DSPELSLMIFISFSPSSQKKIFFNYLYKYYYL-SAFDLKILKPTVLDTEYFLNKLWLNKLTKHPQIPQNILKGQSYGTFRLMRQYSLWAST-AYPKD
KHSTLNISICQSDF-KFIHKS-YNHFFGLFT-YSER-PQRKYTDCDITKKNQ-L-A-TQFITIVS--DINNNLQ-IK-YTVPLNVSDNEMWKRSIIV
IIIIAENYLDIIWFCF-IRKGIN-NGNVTFSSHKPRTVLSEALASLHPSWLQIAKISYK-NSLLFT-NKCILFLKGPFFFSFNLLKRS-IAVLS-GL
PENLQKFR-ENIGRVQRK-TLILKLVKQNRRNEVMRRLCLLAKWLKEEKKRNSMQVARLDSNAPTELASRLR-N-CVK-I-AF-RGIL-GHCYFCNV
IWTAKLNCTTENLRKHIIIQAQLGTSLTS-QNLSTRPAVHKDKKKFASKV-T-I-TIIWFSSLLLLILKVDPFADIDFTCYQPYKQILEYLNKIFLF
-DNTAVAPPP--IWHFDLKGSFNSHTKKVYTLGEKGTSLIKYFNKQL--NFY-FYRFKSKTASFWYMYKFNNFKKSNKLLQVILPYEFRNFKLLFHC
-F-KVDMKLLVMSNLNIHLIMH-YLFLFDDRKITKSTLKFQTLVTSNASWKMHWKHRQCKNFQHFAIL-QCLSIQTSLLSFL-YKLKFK-KDVKTHF
SLVYKTE-KLFSKLNSNSKIFRVVT-KSSQV-LLSPQQEY-LLLQKMLSNIRSLI-ILLNN-INRHGSYK-INMIIRLLQFSRTFF--YCGIDII-S
GAGRTCIDGDCFNELWIWNTE-SKGTALHVIADERLN-GSVDDLKTHFSTKSAWLPVGRRTYNITHKSFQGEWCSNFYHGFTKDGIWPRLFYQKGRI
QLHSIDVTIECFQLCRHFLFSCKFFK-FIGLQHKAFAVLIKRLVFHLPALVNGSL-YFLYCGWNIPTTSNPASNLPGE-FNKLGNSMFQQYRVVVDK
LLIVVFLVMNLKGVRQKGMPVIEGVKFSSNTILVLELFVEK-FWVKFQFEVIATQVLHIIFNHNLNSLSYQY-PACLEQLLITPFRMLLFKHIAYSV
VFPQPDCGIHHQTRHQTKDLLPHSKP-MLRDLRRVDHLCLSIFDCGRIHLSINYLIFHFSILEKAQFTLSSQSFTVNMI-VHPMAQPVGESWPSLMI
CLEKLMREVLGKECKHHV-ALGIRDELSS-KQFFHNLI-IRKLLFTFDTQGSRAIIRHRRKGSSIIGEVREGFPAFVAIVSIVSWWTVNQM-SGCQH
ITHNCISKEGNVIHNVPKSLEVGKYVVHCVWSGFQK-FYSCQKFRC-AVFSIESCPGTIALILVKSHTHTYTSVLTWVGTARICCMAERQDRAQQQG
AGEHRSGGQGAARG-ALLSEGVWLWS-RRRC-GVPGRAPDAHCKSYDN
```



3'5' Frame 2

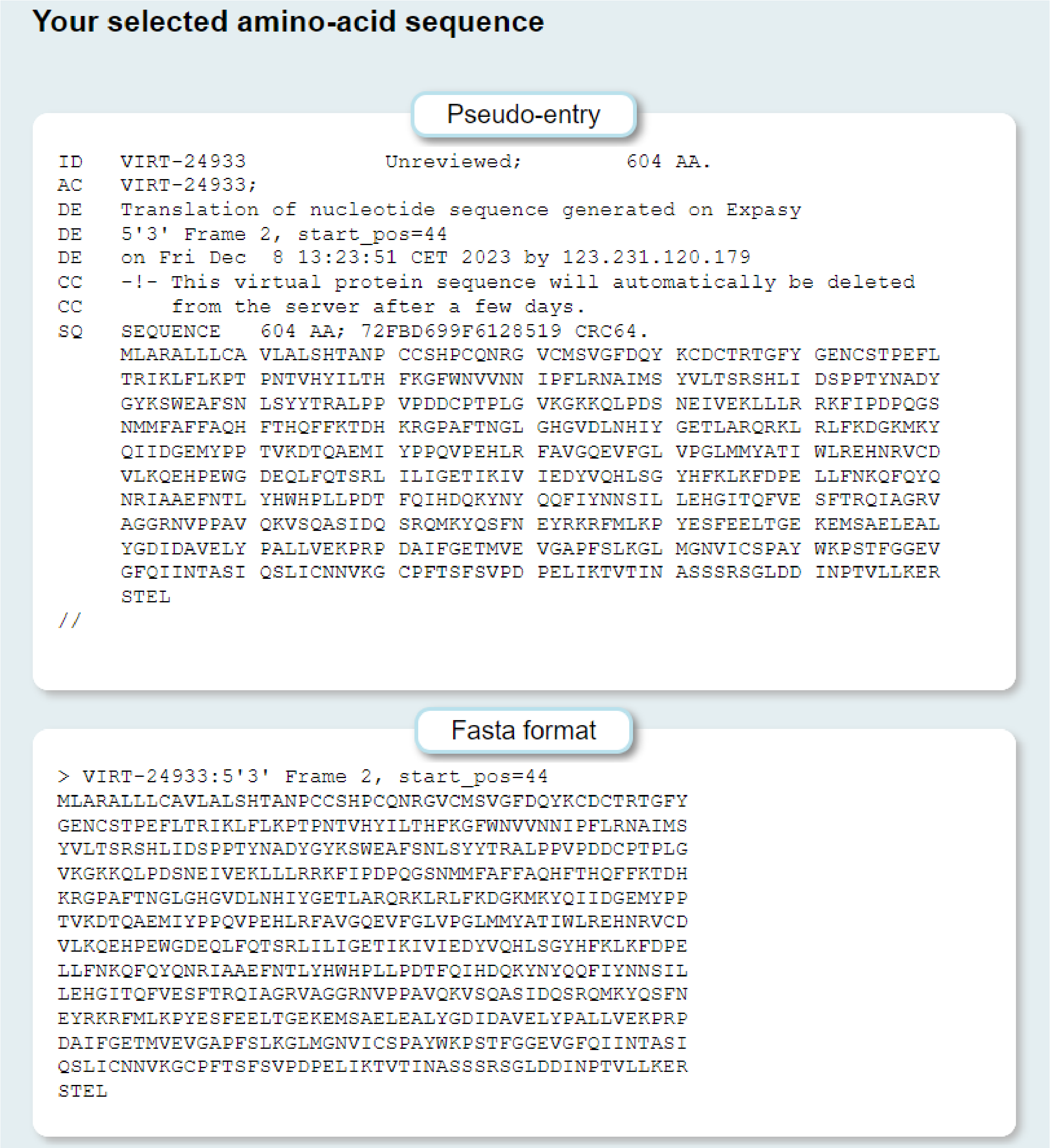
```
IETIKVYLQKRSVNF-IVQK-TDIYFQTYLLHKLNTHLSEALKHSHTQPKFPGFVSRSMLITNQVFFLKKTVNSVLYKHIDYFFLNLKHSKRKIVKK
ILLSYL--YLFHFLPLPKRKFFSIIYINTIICNQRLI-KY-NPQCLTQNIFSINCG-TN-RSIHRSLKTF-KVSLMAHSG--DNILCGLAHRPILRT
NIQP-ISQYANQISKNLFTNLNTIISLDCLHNIQRGNPKENILIVT-QKKINDCRLKHSL-P-CPNKT-IIIFNKLSNIQYH-MSVTMRCGKEVLSS
LLLLLRTT-ISFGFVFK-ERALIRMGT-LFPVTNPVQFSLRH-PLCIHLGYR-LKLAINKILCFSLKISVYYF-KGLFFSLLIFLRGAK-QS-AEVY
LKTCKSSGKKI-AESKESEL-S-N--NKIGETK--EDCVS-QNG-KKKRKGTACR-PGLIPMHLLNWLQD-DKINV-NEFKHFEEVFYEVIATFAM-
YGLLN-TVQQKI-ENISLFKHSLVLH-PHSKI-VPGLQCTRIKKSLLQKFKPKFEQ-FGFLLCYCLF-R-TLLLI-ILHVTSHINKFWNISTKYSYF
KITLQWLHHLNKFGILI-KEVLTVTLKRFILLVKKEHHL-NILTSNCSKISTDFTDLNPRQLLFGICTSLITLRNQTSFYR-FYPMNLEISNYCFIA
DFKK-T-NYW-CLI-IFI--CTDTCFCLMTEK-PKVL-NFKH-LLLMHHGRCIGNIDSVKTFSILPSCDSV-VFKHLYFRSYNIS-NSSKKTSKLIS
LWFIKQNKNCFPNLTATAKSLE--HKSLHKYDSFLRNRSTDFCYRRC-VT-GV-YKYY-IIKLIDMVHINK-I-SLDFYSSVERSFSSTVGLISSSP
EREELALMVTVLMSSGSGTLNEVKGQPFTLLQMRD-IEAVLMI-KPTSPPKVLGFQ-AGEHITLPISPFKENGAPTSTMVSPKMASGRGFSTRRAGY
SSTASMSP-SASNSADISFSPVSSSNDS-GFSIKRLRYSLKDWYFICLLWSMEACDTFCTAGGTFLPPATLPAICLVNDSTNWVIPCSSNIELL-IN
C--LYFWS-I-KVSGRRGCQ--RVLNSAAIRFWYWNCLLKSSSGSNFSLK--PLKCCT-SSITILIVSPISISLLVWNNCSSPHSGCSCLSTSHTLL
CSRSQIVAYIIRPGTRPKTSCPTANRRCSGT-GG-IISA-VSLTVGGYISPSII-YFIFPSLKRRSLRCLARVSP-I-FKSTPWPSPLVKAGPRL-S
VLKN-CVKCWAKNANIMFEPWGSGMNFLLRSNFSTISFESGSCFLPLTPKGVGQSSGTGGRALV--ERLEKASQLL-P-SAL-VGGLSIKCDLDVNT
-LIIAFRRKGMLFTTFQNPLKWVSM-CTVFGVGFRNNFILVKNSGVEQFSP-NPVRVQSHLYWSNPTLIHTPRF-HGWEQQGFAVWLSARTAHSSRA
RASIAAAGRARRGGRLCCLRASGCGAEGGAAEEFLDVLLTLTASRMTI
```



3'5' Frame 3

```
-KQSKFIFRKDLSIFK-YKNKQTFISRLIFYIS-IHICLRH-NIRIHNPNSQVLSADQC--QIKFFF-KKQ-TVFYTSILTISFLTLNILNVKL-RR
FS-VIFNDIYFIFSLFPKENFFQLFI-ILLSVISV-FKNIKTHSA-HRIFSQ-IVAEQINEASTDPSKHFKRSVLWHIQANETIFFVG-HIGLS-GQ
TFNLKYLNMPIRFLKIYSQILIQSFLWIVYIIFREVTPKKIY-L-HNKKKSMIVGLNTVYNHSVLIRHK--SSIN-VIYSTIKCQ-Q-DVEKKYYRH
YYYC-ELLRYHLVLFLNKKGH-LEWERNFFQSQTPYSSL-GTSLFASILVTDS-N-L-IKFSAFHLK-VYTIFKRAFFFLF-SS-EELNSSPELRFT
-KLAKVQVRKYRQSPKKVNSDLKTSKTK-EKRSDEKTVSLSKMAKRRKEKEQHAGSQA-FQCTY-IGFKTEIKLMCEMNLSILKRYFMRSLLLLQCD
MDC-IKLYNRKSEKTYHYSSTAWYFINLIAKSEYQACSAQG-KKVCFKSLNLNLNNNLVFFFVIAYFKGRPFC-YRFYMLPAI-TNSGISQQNILIL
R-HCSGSTTLINLAF-FKRKF-QSH-KGLYSW-KRNITYKIF-QAIVVKFLLILQI-IQDSFFLVYVQV--L-EIKQAFTGDSTL-I-KFQIIVSLL
ILKSRHEITGNV-FKYSFNNALIPVFV--QKNNQKYFKISNISYF-CIMEDALETSTV-KLSAFCHLVTVFKYSNIFTFVLII-VEIQVKRRQNSFL
SGL-NRIKTVFQT-QQQQNL-SSDIKVFTSMTPFSATGVLTSVTEDVK-HKEFNINIIK-LN--TWFI-INKYDH-TSTVQSNVLLVVLWD-YHLVR
SGKNLH-W-LF--ALDLEH-MK-RDSPSRYCR-ETELRQC--FENPLLHQKCLASSRQENI-HYP-VLSRRMVLQLLPWFHQRWHLAEAFLPEGQDT
APQHRCHHRVLPTLQTFPFLL-VLQMIHRASA-SVCGTH-KTGISSACSGQWKPVILSVLRVEHSYHQQPCQQFAW-MIQQTG-FHVPAI-SCCR-T
VDSCISGHEFERCQAEGDASDRGC-IQQQYDFGTGIVC-KVVLGQISV-SDSHSSVAHNLQSQS--SLLSVLACLSGTTAHHPIQDAPV-AHRILCC
VPAARLWHTSSDQAPDQRPPAPQQTVDAQGLEEGRSSLPEYL-LWEDTSLHQLSDISFFHP-KGAVYAV-PEFHRKYDLSPPHGPARW-KLALAYDL
S-KTDA-SAGQRMQTSCLSPGDQG-TFFLEAIFPQSHLNQEAAFYL-HPRESGNHQAQEEGL-YNRRG-RRLPSFCSHSQHCKLVDCQSNVIWMSTH
NS-LHFEGRECYSQRSKIP-SG-VCSALCLEWVSEIILFLSKIPVLSSFLHRILSGYNRTYTGQIPHSYIHLGFDMGGNSKDLLYG-APGPRTAAGR
GRASQRRAGRGAGVGFAV-GRLAVELKEALLRSSWTCS-RSLQVV-Q
```

The selected sequence can be viewed by clicking on the first amino acid (M) of the sequence.



Your selected amino-acid sequence

Pseudo-entry

```
ID   VIRT-24933             Unreviewed;        604 AA.
AC   VIRT-24933;
DE   Translation of nucleotide sequence generated on Expasy
DE   5'3' Frame 2, start_pos=44
DE   on Fri Dec  8 13:23:51 CET 2023 by 123.231.120.179
CC   -!- This virtual protein sequence will automatically be deleted
CC       from the server after a few days.
SQ   SEQUENCE   604 AA; 72FBD699F6128519 CRC64.
     MLARALLLCA VLALSHTANP CCSHPCQNRG VCMSVGFDQY KCDCTRTGFY GENCSTPEFL
     TRIKLFLKPT PNTVHYILTH FKGFWNVVNN IPFLRNAIMS YVLTSRSHLI DSPPTYNADY
     GYKSWEAFSN LSYYTRALPP VPDDCPTPLG VKGKKQLPDS NEIVEKLLLR RKFIPDPQGS
     NMMFAFFAQH FTHQFFKTDH KRGPAFTNGL GHGVDLNHIY GETLARQRKL RLFKDGKMKY
     QIIDGEMYPP TVKDTQAEMI YPPQVPEHLR FAVGQEVFGL VPGLMMYATI WLREHNRVCD
     VLKQEHPEWG DEQLFQTSRL ILIGETIKIV IEDYVQHLSG YHFKLKFDPE LLFNKQFQYQ
     NRIAAEFNTL YHWHPLLPDT FQIHDQKYNY QQFIYNNSIL LEHGITQFVE SFTRQIAGRV
     AGGRNVPPAV QKVSQASIDQ SRQMKYQSFN EYRKRFMLKP YESFEELTGE KEMSAELEAL
     YGDIDAVELY PALLVEKPRP DAIFGETMVE VGAPFSLKGL MGNVICSPAY WKPSTFGGEV
     GFQIINTASI QSLICNNVKG CPFTSFSVPD PELIKTVTIN ASSSRSGLDD INPTVLLKER
     STEL
//
```

Fasta format

```
> VIRT-24933:5'3' Frame 2, start_pos=44
MLARALLLCAVLALSHTANPCCSHPCQNRGVCMSVGFDQYKCDCTRTGFY
GENCSTPEFLTRIKLFLKPTPNTVHYILTHFKGFWNVVNNIPFLRNAIMS
YVLTSRSHLIDSPPTYNADYGYKSWEAFSNLSYYTRALPPVPDDCPTPLG
VKGKKQLPDSNEIVEKLLLRRKFIPDPQGSNMMFAFFAQHFTHQFFKTDH
KRGPAFTNGLGHGVDLNHIYGETLARQRKLRLFKDGKMKYQIIDGEMYPP
TVKDTQAEMIYPPQVPEHLRFAVGQEVFGLVPGLMMYATIWLREHNRVCD
VLKQEHPEWGDEQLFQTSRLILIGETIKIVIEDYVQHLSGYHFKLKFDPE
LLFNKQFQYQNRIAAEFNTLYHWHPLLPDTFQIHDQKYNYQQFIYNNSIL
LEHGITQFVESFTRQIAGRVAGGRNVPPAVQKVSQASIDQSRQMKYQSFN
EYRKRFMLKPYESFEELTGEKEMSAELEALYGDIDAVELYPALLVEKPRP
DAIFGETMVEVGAPFSLKGLMGNVICSPAYWKPSTFGGEVGFQIINTASI
QSLICNNVKGCPFTSFSVPDPELIKTVTINASSSRSGLDDINPTVLLKER
STEL
```

The selected result is similar to the Homo sapiens PTGS2 reference protein sequence retrieved from NCBI.

Readings:

http://www.aspree.org/AUS/aspree-content/aspirin/how-aspirin-works.aspx

Q2. Python Exercises

1. Below shows some files with embedded sample names:

```
lane1_NewCode_L001_R1.fastq.gz
lane1_NoIndex_L001_R1.fastq.gz
lane1_NoIndex_L001_R2.fastq.gz
pipeline_processing_output.log
lane7027_ACTGAT_JH25_L001_R1.fastq.gz
lane7027_ACTTGA_E30_1_2_Hap4_24h_L001_R1.fastq.gz
lane7027_AGTTCC_JH14_L001_R1.fastq.gz
lane7027_CGGAAT_JH37_L001_R1.fastq.gz
lane7027_GCCAAT_E30_1_2l_Hap4_log_L001_R1.fastq.gz
lane7127_GGCTAC_E30_1_4_Hap4_48h_L001_R1.fastq.gz
```

Write a Python code to extract the sample name from these files ignoring any files which do not match the format given below.

The format is:

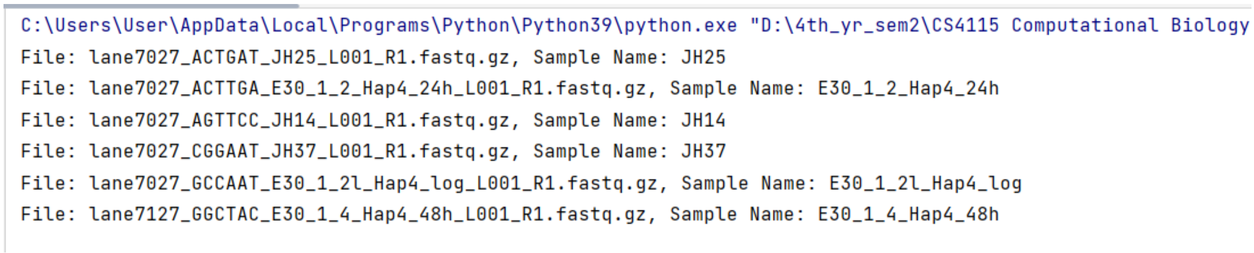
1. Written lane number
2. Barcode
3. Sample name
4. Numeric lane number (starting with L)
5. Read number (R1/2/3/4)
6. File extension

Eg. `Lane8127_GCCAAT_S30_1_2l_Hap4_log_L001_R1.fastq.gz` the sample name would be,

`S30_1_2l_Hap4_log`

```
'''
Extract sample names
08/12/2023
Nimna Gamage
s14682
Lab 01-Question2_Sub_question1
'''

#method 1

#define function
def extract_sample_name(file_name):
    # Split the file name using underscores
    parts = file_name.split('_')

    # Check the file name for the expected format
    if len(parts) >= 5 and parts[0].startswith('lane') and parts[-
1].endswith('.fastq.gz'):
        # Extract the sample name from the appropriate position
        sample_name = parts[2:-2]

        # Join parts[2:] to handle cases where the sample name contains
underscores
        return '_'.join(sample_name)
    else:
        return None


# List of given file names
file_names = [
    "lane1_NewCode_L001_R1.fastq.gz",
    "lane1_NoIndex_L001_R1.fastq.gz",
    "lane1_NoIndex_L001_R2.fastq.gz",
    "pipeline_processing_output.log",
    "lane7027_ACTGAT_JH25_L001_R1.fastq.gz",
    "lane7027_ACTTGA_E30_1_2_Hap4_24h_L001_R1.fastq.gz",
    "lane7027_AGTTCC_JH14_L001_R1.fastq.gz",
    "lane7027_CGGAAT_JH37_L001_R1.fastq.gz",
    "lane7027_GCCAAT_E30_1_21_Hap4_log_L001_R1.fastq.gz",
    "lane7127_GGCTAC_E30_1_4_Hap4_48h_L001_R1.fastq.gz"
]

# Extract and print sample names
for file_name in file_names:
    sample_name = extract_sample_name(file_name)
    if sample_name is not None:
        print(f"File: {file_name}, Sample Name: {sample_name}")
```

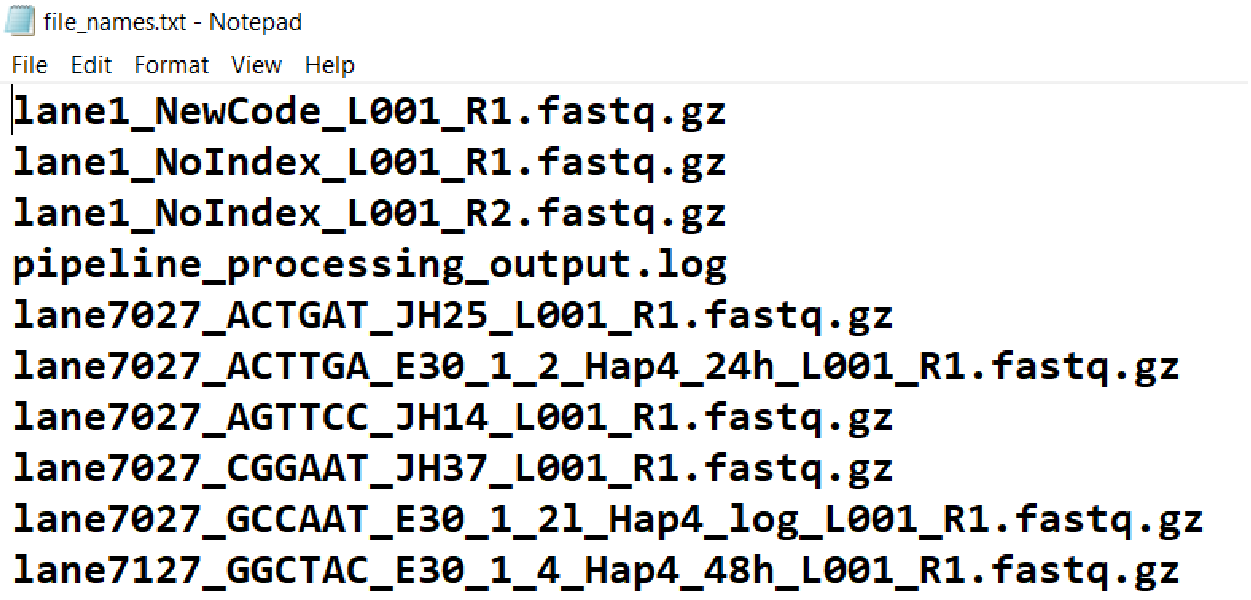
Output:



```
C:\Users\User\AppData\Local\Programs\Python\Python39\python.exe "D:\4th_yr_sem2\CS4115 Computational Biology
File: lane7027_ACTGAT_JH25_L001_R1.fastq.gz, Sample Name: JH25
File: lane7027_ACTTGA_E30_1_2_Hap4_24h_L001_R1.fastq.gz, Sample Name: E30_1_2_Hap4_24h
File: lane7027_AGTTCC_JH14_L001_R1.fastq.gz, Sample Name: JH14
File: lane7027_CGGAAT_JH37_L001_R1.fastq.gz, Sample Name: JH37
File: lane7027_GCCAAT_E30_1_2l_Hap4_log_L001_R1.fastq.gz, Sample Name: E30_1_2l_Hap4_log
File: lane7127_GGCTAC_E30_1_4_Hap4_48h_L001_R1.fastq.gz, Sample Name: E30_1_4_Hap4_48h
```

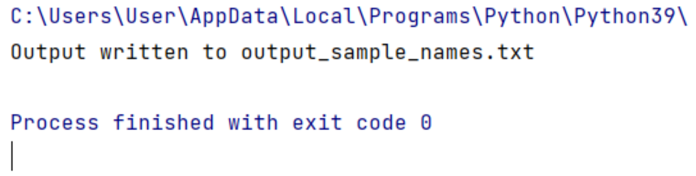
```
#method 2 - using regular expressions

import re

#define function
def extract_sample_name(file_name):
    # Define the pattern for matching the desired format
    pattern = re.compile(r'^lane\d+_[A-Z0-9]+_([A-Za-z0-
9_]+)_L\d+_[R]\d\.fastq\.gz$')

    # Attempt to match the pattern with the file name
    match = pattern.match(file_name)

    # If there is a match, return the extracted sample name
    if match:
        return match.group(1)
    else:
        return None

# Read file names from a text file
file_names_file = "file_names.txt"
output_file_path = "output_sample_names.txt"

#open the text file
with open(file_names_file, "r") as file:
    file_names = file.read().splitlines()

# Extract and write sample names to an output file
with open(output_file_path, "w") as output_file:
    for file_name in file_names:
        sample_name = extract_sample_name(file_name)
        if sample_name is not None:
            output_file.write(f"File: {file_name}, Sample Name:
{sample_name}\n")

print("Output written to", output_file_path)
```

Input file:

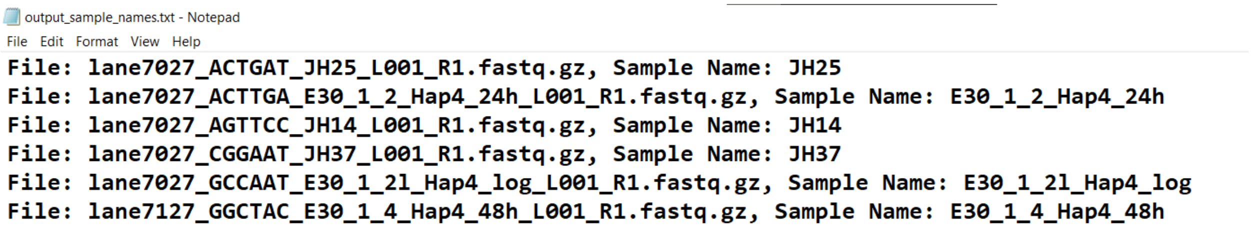


```
lane1_NewCode_L001_R1.fastq.gz
lane1_NoIndex_L001_R1.fastq.gz
lane1_NoIndex_L001_R2.fastq.gz
pipeline_processing_output.log
lane7027_ACTGAT_JH25_L001_R1.fastq.gz
lane7027_ACTTGA_E30_1_2_Hap4_24h_L001_R1.fastq.gz
lane7027_AGTTCC_JH14_L001_R1.fastq.gz
lane7027_CGGAAT_JH37_L001_R1.fastq.gz
lane7027_GCCAAT_E30_1_21_Hap4_log_L001_R1.fastq.gz
lane7127_GGCTAC_E30_1_4_Hap4_48h_L001_R1.fastq.gz
```

Output;



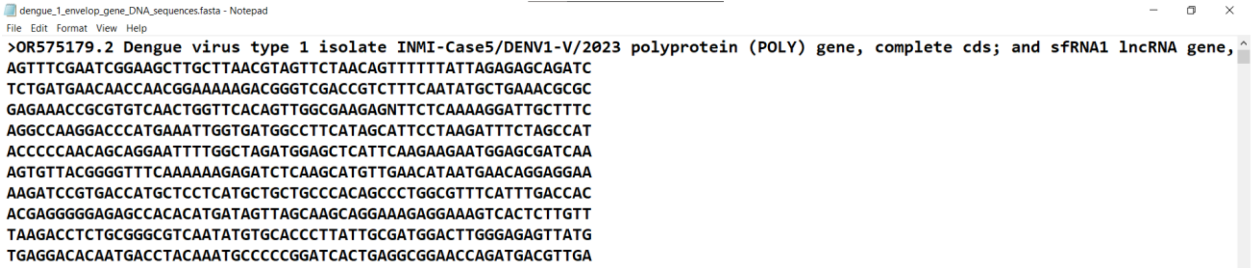
```
C:\Users\User\AppData\Local\Programs\Python\Python39\
Output written to output_sample_names.txt

Process finished with exit code 0
```



```
File: lane7027_ACTGAT_JH25_L001_R1.fastq.gz, Sample Name: JH25
File: lane7027_ACTTGA_E30_1_2_Hap4_24h_L001_R1.fastq.gz, Sample Name: E30_1_2_Hap4_24h
File: lane7027_AGTTCC_JH14_L001_R1.fastq.gz, Sample Name: JH14
File: lane7027_CGGAAT_JH37_L001_R1.fastq.gz, Sample Name: JH37
File: lane7027_GCCAAT_E30_1_21_Hap4_log_L001_R1.fastq.gz, Sample Name: E30_1_21_Hap4_log
File: lane7127_GGCTAC_E30_1_4_Hap4_48h_L001_R1.fastq.gz, Sample Name: E30_1_4_Hap4_48h
```

2. Create a FASTA file by obtaining 10 Dengue 1- Envelop gene DNA sequences from NCBI. Write a Python-program that reads the FASTA file, cleans up the header line to have only Accession number & gene-name and print headers and sequences to standard output as multi-FASTA-file again.

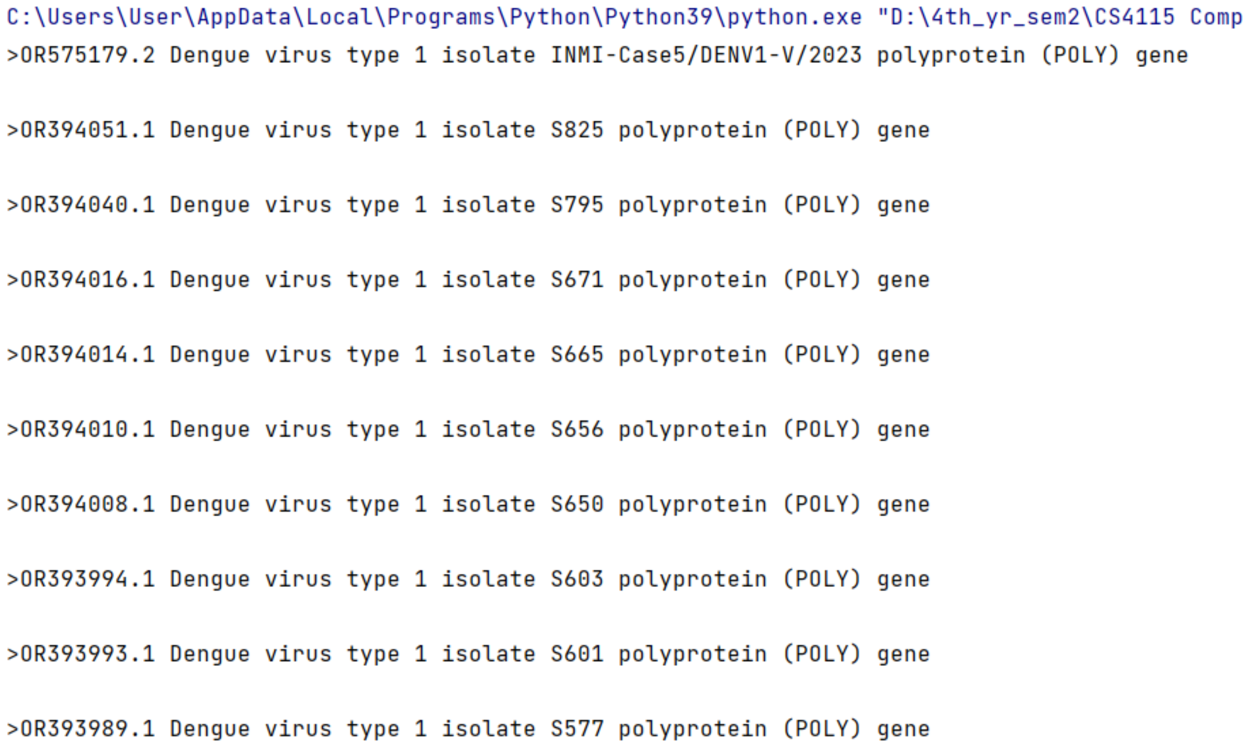
Created fasta file;


dengue_1_envelop_gene_DNA_sequences.fasta - Notepad
File Edit Format View Help

```
>OR575179.2 Dengue virus type 1 isolate INMI-Case5/DENV1-V/2023 polyprotein (POLY) gene, complete cds; and sfRNA1 lncRNA gene,
AGTTTCGAATCGGAAGCTTGCTTAACGTAGTTCTAACAGTTTTTTATTAGAGAGCAGATC
TCTGATGAACAACCAACGGAAAAAGACGGGTCGACCGTCTTTCAATATGCTGAAACGCGC
GAGAAACCGCGTGTCAACTGGTTCACAGTTGGCGAAGAGNTTCTCAAAAGGATTGCTTTC
AGGCCAAGGACCCATGAAATTGGTGATGGCCTTCATAGCATTCCTAAGATTTCTAGCCAT
ACCCCCAACAGCAGGAATTTTGGCTAGATGGAGCTCATTCAAGAAGAATGGAGCGATCAA
AGTGTTACGGGGTTTCAAAAAAGAGATCTCAAGCATGTTGAACATAATGAACAGGAGGAA
AAGATCCGTGACCATGCTCCTCATGCTGCTGCCCACAGCCCTGGCGTTTCATTTGACCAC
ACGAGGGGGAGAGCCACACATGATAGTTAGCAAGCAGGAAAGAGGAAAGTCACTCTTGTT
TAAGACCTCTGCGGGCGTCAATATGTGCACCCTTATTGCGATGGACTTGGGAGAGTTATG
TGAGGACACAATGACCTACAAATGCCCCCGGATCACTGAGGCGGAACCAGATGACGTTGA
```

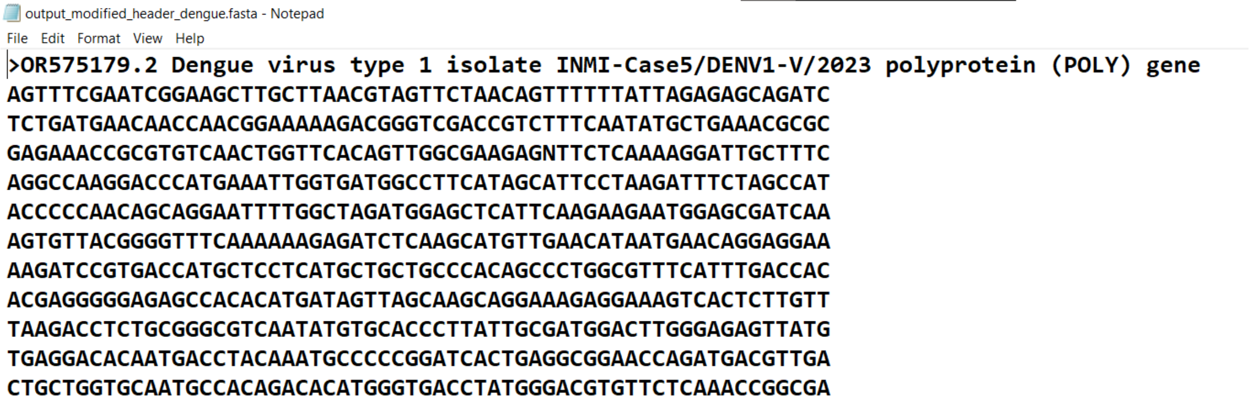
```python
# '''
# modify fasta header
# 08/12/2023
# Nimna Gamage
# s14682
# Lab 01-Question2_Sub_question2
# '''

#define function
def modify_fasta_header(input_fasta, output_fasta):
    with open(input_fasta, 'r') as input_file, open(output_fasta, 'w') as
output_file:
        for line in input_file:
            if line.startswith('>'):
                # Extract the part before the first comma
                header = line.strip().split(',')[0]
                output_file.write(f'{header}\n')
                print(f'{header}\n')
            else:
                output_file.write(line)

modify_fasta_header('dengue_1_envelop_gene_DNA_sequences.fasta',
'output_modified_header_dengue.fasta')
```

Console output;



```
C:\Users\User\AppData\Local\Programs\Python\Python39\python.exe "D:\4th_yr_sem2\CS4115 Comp
>OR575179.2 Dengue virus type 1 isolate INMI-Case5/DENV1-V/2023 polyprotein (POLY) gene

>OR394051.1 Dengue virus type 1 isolate S825 polyprotein (POLY) gene

>OR394040.1 Dengue virus type 1 isolate S795 polyprotein (POLY) gene

>OR394016.1 Dengue virus type 1 isolate S671 polyprotein (POLY) gene

>OR394014.1 Dengue virus type 1 isolate S665 polyprotein (POLY) gene

>OR394010.1 Dengue virus type 1 isolate S656 polyprotein (POLY) gene

>OR394008.1 Dengue virus type 1 isolate S650 polyprotein (POLY) gene

>OR393994.1 Dengue virus type 1 isolate S603 polyprotein (POLY) gene

>OR393993.1 Dengue virus type 1 isolate S601 polyprotein (POLY) gene

>OR393989.1 Dengue virus type 1 isolate S577 polyprotein (POLY) gene
```

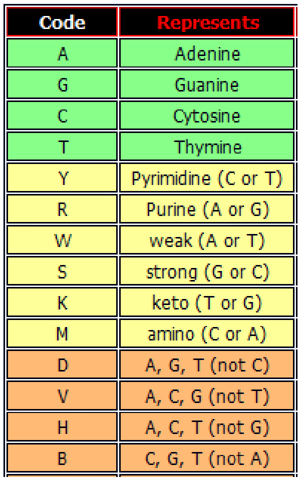
Output file; header contain only the accession number and gene name



output_modified_header_dengue.fasta - Notepad
File Edit Format View Help

```
>OR575179.2 Dengue virus type 1 isolate INMI-Case5/DENV1-V/2023 polyprotein (POLY) gene
AGTTTCGAATCGGAAGCTTGCTTAACGTAGTTCTAACAGTTTTTTATTAGAGAGCAGATC
TCTGATGAACAACCAACGGAAAAAGACGGGTCGACCGTCTTTCAATATGCTGAAACGCGC
GAGAAACCGCGTGTCAACTGGTTCACAGTTGGCGAAGAGNTTCTCAAAAGGATTGCTTTC
AGGCCAAGGACCCATGAAATTGGTGATGGCCTTCATAGCATTCCTAAGATTTCTAGCCAT
ACCCCCAACAGCAGGAATTTTGGCTAGATGGAGCTCATTCAAGAAGAATGGAGCGATCAA
AGTGTTACGGGGTTTCAAAAAAGAGATCTCAAGCATGTTGAACATAATGAACAGGAGGAA
AAGATCCGTGACCATGCTCCTCATGCTGCTGCCCACAGCCCTGGCGTTTCATTTGACCAC
ACGAGGGGGAGAGCCACACATGATAGTTAGCAAGCAGGAAAGAGGAAAGTCACTCTTGTT
TAAGACCTCTGCGGGCGTCAATATGTGCACCCTTATTGCGATGGACTTGGGAGAGTTATG
TGAGGACACAATGACCTACAAATGCCCCCGGATCACTGAGGCGGAACCAGATGACGTTGA
CTGCTGGTGCAATGCCACAGACACATGGGTGACCTATGGGACGTGTTCTCAAACCGGCGA
```

3. Write a Python program to search the DNA Sequence for the presence of one of the following Transcription Factor Binding Sites(TFBS) with ambiguity codes. Search for all the positions in the sequence where TFBS is located.



Code	Represents
A	Adenine
G	Guanine
C	Cytosine
T	Thymine
Y	Pyrimidine (C or T)
R	Purine (A or G)
W	weak (A or T)
S	strong (G or C)
K	keto (T or G)
M	amino (C or A)
D	A, G, T (not C)
V	A, C, G (not T)
H	A, C, T (not G)
B	C, G, T (not A)

Transcription Factor	Consensus Sequence
RUNX1	BHTGTGGTYW
TGIF1	WGACAGB
IKZF1	BTGGGARD

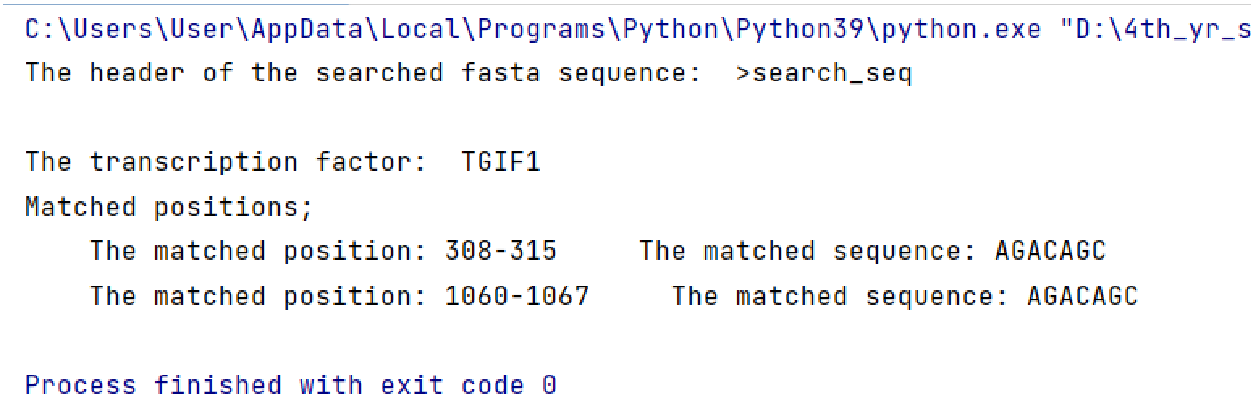
The sequence is shown below.

```
>search_seq
GACACCTCAGTACTAGGATGNNNNNNTATCAGCCTGAACTAGCAGGCCTGGTTCCAAATT
TTTTTATCAACACTCGTAGGGGGATTATCCTAGAGGGGGTCTGGGATTTCTTTGACATCA
GAGTATTTTTGCCTTGCTCCTTCACAATTTGGGAACAAATAATTTAGTGGTTATTAACCC
TGGCTACGCACTGGAAACTTTAAAAATAATGCTGGTATGAAATTTACACAGAGTATCGTG
AAAATTTTCACTGAGTACCATGTGGTTATACATTGGATAAGGCTCCAGGAAGCAGCTACT
GGAAGACAGCCATGCCAAGAGTGGTTAGTGGTTGGAATTTTGGCAAGTCAGTTTTAGTCT
GCCTTATCAAATACATGGGCATACAGATAAATCCTTAGATGGCTCTCCTACTTACTGAAA
CATTTTCTATCTATCTATCTATCTATCTATCTATTTGGGAAGCTATCTATCTATCTATCA
TTTATTTAAGGTAGTCTCTATCTGCCTCTGTCTCTGTCTGTCTCTGTGTCTCTGTGTCTG
TCTGCTCTCTCTCTCTCTCTGTGGGAATCTCTCTCTGTGTGTGTGTGTGTATGTGTGTGT
GTGTGTGTGTGGTGTGCATGAACATGAGTAAAATCCATAAGGAAACTTTCAGAGTTGGTC
CTCTCCTTATATCAAATGGATCCAGGAATTAAACTCAGGTTCAATTCTTGGTGCCTTTAC
TAGTTGAGCCATCTCACTGGCTCTTCATCATCTTTAGAATAAACTCACTTTATTACACAC
ACACACACACACACAACCTGGGAGTACACACACACACACAACCAAAGCCCCAACGGAAAA
CTACAATATTATAATGAATACACAGGTTCTCAACATAGTCTCTGCCACGCTTGCAGACAA
AGATGAGTAGAAGTAGAAAGAACCAGGGAAACGTGGAGCAAGTCAGAAGGAATAACAGTC
AGAAGGAATAACAGTCAGAAGGAATAACAGTCAGAAGGAGTAACAGTCAGAAGGAATAGC
AGTCAGAAGGAATAACAGTCAGAAGACAGCACAGTCAGAAGGAATAACAGTCAGAAGGAA
```

```
TAACAGTCAGAAGGAATAACAGTCAGAAGGAATAACAGTCAGAAGGAATAGCAGTCAGAA

GGAATAACAGTCAGAAGGAATAACAGTCAGAAGGAATAACAGTCAAAGAAATAGCAGTCA

GAAGGAATAGCAGTCAGAAGGAATAACAGTCAAAGGAGCAGTCAGAAGGAGTAACAGTCA

GAAGGAATAACAGTCAGAAGGAATAACAGTCAAAGGAATAGCAGTCAGAAGGAGTAACAG

TCAGAGCAAACACAGAGATGACAAAGGCAATGGGGTCAGAGACTTCACCACTCTCCAAGA
```

```
# '''
# predict TFBS
# 08/12/2023
# Nimna Gamage
# s14682
# Lab 01-Question2_Sub_question3
# '''

#method 1

import re

filename = "search_seq.fasta"
tf = "TGIF1"
header = ""
sequence = ""

with open(filename, 'r') as file:
    for line in file:
        if line != '\n':
            if '>' in line:
                 header += line
            else:
                sequence += line


matches = re.finditer("[AT]GACAG[CGT]", sequence)

print("The header of the searched fasta sequence: ", header)
print("The transcription factor: ", tf)
print("Matched positions; ")

for match in matches:
    start, end = match.start(), match.end()
    matched_sequence = sequence[start:end]
    print(f"    The matched position: {start}-{end}     The matched sequence:
{matched_sequence}")
```

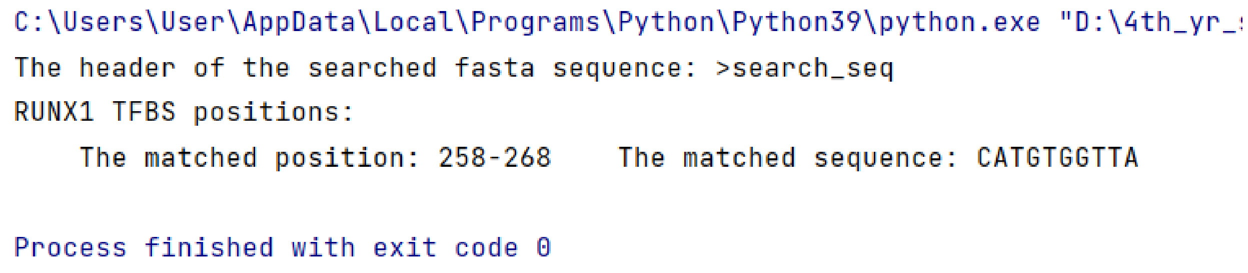
Output;



```
C:\Users\User\AppData\Local\Programs\Python\Python39\python.exe "D:\4th_yr_s
The header of the searched fasta sequence:  >search_seq

The transcription factor:  TGIF1
Matched positions;
    The matched position: 308-315     The matched sequence: AGACAGC
    The matched position: 1060-1067     The matched sequence: AGACAGC

Process finished with exit code 0
```

```
###method 2

#define function
def read_code_representations(code_file):
    code_representations = {}
    with open(code_file, 'r') as file:
        for line in file:
            parts = line.strip().split()
            if len(parts) >= 2:
                code, description = parts[0], ' '.join(parts[1:])
                code_representations[code] = description
    return code_representations

#define function
def search_tfbs(sequence, tfbs, code_representations):
    positions = []
    for i in range(len(sequence) - len(tfbs) + 1):
        match = True
        for j in range(len(tfbs)):
            current_nucleotide = sequence[i + j]
            if current_nucleotide not in code_representations[tfbs[j]]:
                match = False
                break
        if match:
            positions.append((i, i + len(tfbs)))
    return positions

##main
if __name__ == "__main__":
    code_file = "code_represents.txt"
    sequence_file = "search_seq.fasta"

    #transcription factor binding sites
    tfbs_name = "RUNX1"
    tfbs_sequence = "BHTGTGGTYW"
    #code representations
    code_representations = read_code_representations(code_file)

    # Read DNA sequence by opening file
    with open(sequence_file, 'r') as file:
        lines = file.readlines()
        header = lines[0].strip()
        sequence = "".join(line.strip() for line in lines[1:])

    # Search for TFBS positions with the binding site pattern for RUNX1
    tfbs_positions = search_tfbs(sequence, tfbs_sequence,
code_representations)

    # Print results
    print(f"The header of the searched fasta sequence: {header}")
    print(f"{tfbs_name} TFBS positions:")
    for start, end in tfbs_positions:
        binding_site = sequence[start:end]
        print(f"    The matched position: {start}-{end}    The matched
sequence: {binding_site}")
```

Output:



```
C:\Users\User\AppData\Local\Programs\Python\Python39\python.exe "D:\4th_yr_
The header of the searched fasta sequence: >search_seq
RUNX1 TFBS positions:
    The matched position: 258-268    The matched sequence: CATGTGGTTA

Process finished with exit code 0
```

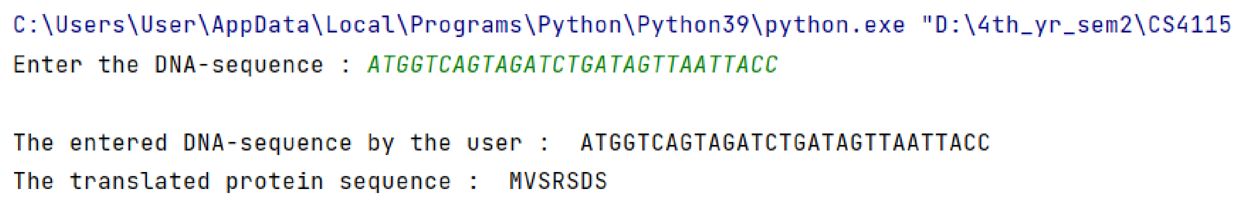
Q3 – Biopython

Biopython Tutorial and Cookbook https://biopython.org/DIST/docs/tutorial/Tutorial.html#sec2

1. Write a Biopython program that asks the user to input a DNA-sequence and then translates the sequence to protein sequence.

```
'''
asks the user to input a DNA-sequence and then translates the sequence to
protein sequence
Input: DNA-sequence
Output: The translated protein sequence
08/12/2023
Nimna Gamage
s14682
Lab 01-Question3_Sub_question1
'''

#import biopython sub-module
from Bio.Seq import Seq

#get the user input
dna_seq = input("Enter the DNA-sequence : ")

#create the biopython Seq object from the entered sequence
dna_sequence = Seq(dna_seq)

#translate the DNA sequence to protein sequence until it encounters a stop
codon
protein_sequence = dna_sequence.translate(to_stop=True)

#print the output
print("\nThe entered DNA-sequence by the user : ", dna_seq)
print("The translated protein sequence : ", protein_sequence)
```

Output;

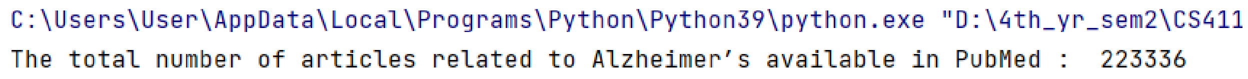


```
C:\Users\User\AppData\Local\Programs\Python\Python39\python.exe "D:\4th_yr_sem2\CS4115
Enter the DNA-sequence : ATGGTCAGTAGATCTGATAGTTAATTACC

The entered DNA-sequence by the user :  ATGGTCAGTAGATCTGATAGTTAATTACC
The translated protein sequence :  MVSRSDS
```

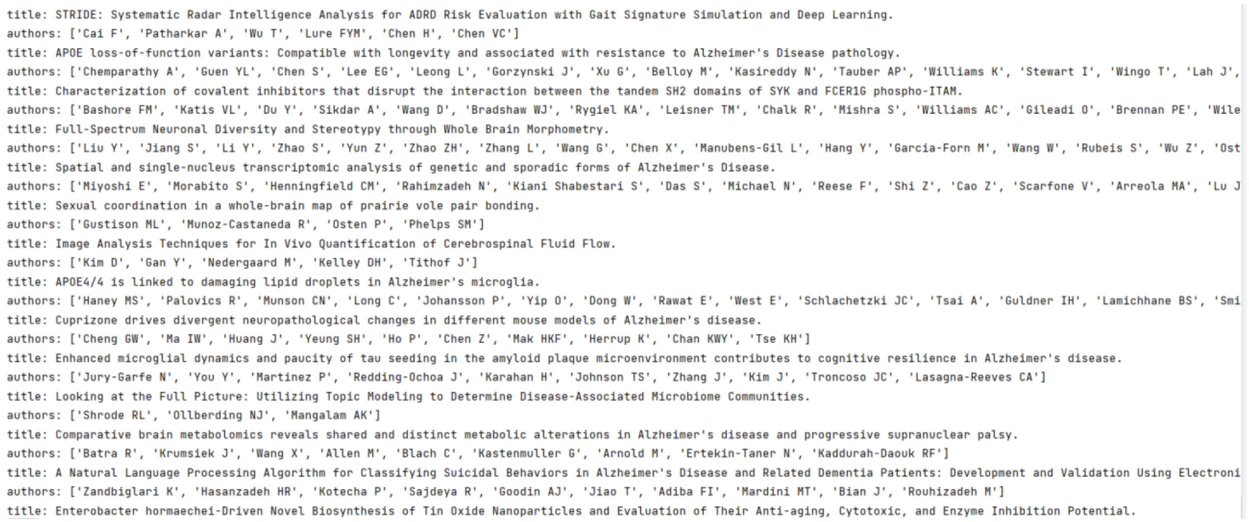
2. Write a Biopython program that will find all articles related to Alzheimer’s in PubMed. Print the total number of articles available and the authors.

```
'''
Find all articles related to Alzheimer’s in PubMed and print the total number
of articles available and the authors
Output: The total number of articles related to Alzheimer’s available in
PubMed and the authors
08/12/2023
Nimna Gamage
s14682
Lab 01-Question3_Sub_question2
'''

#import biopython sub-module
from Bio import Entrez

#provide email address to NCBI
Entrez.email = "nimnagamage65@gmail.com"

#query PubMed for all articles having to do with 'Alzheimer’s'
#checking how many such articles are there
handle = Entrez.egquery(term="Alzheimer’s")
record = Entrez.read(handle)
for row in record["eGQueryResult"]:
    if row["DbName"] == "pubmed":
        print("The total number of articles related to Alzheimer’s available
in PubMed : ", row["Count"])

        handle = Entrez.esearch(db="pubmed", term="Alzheimer’s",
retmax=row["Count"])
        record = Entrez.read(handle)
        handle.close()
        idlist = record["IdList"]


from Bio import Medline
handle = Entrez.efetch(db="pubmed", id=idlist, rettype="medline",
retmode="text")
records = Medline.parse(handle)
records = list(records)
for record in records:
    print("title:", record.get("TI", "?"))
    print("authors:", record.get("AU", "?"))
```

Output;



```
C:\Users\User\AppData\Local\Programs\Python\Python39\python.exe "D:\4th_yr_sem2\CS411
The total number of articles related to Alzheimer’s available in PubMed :  223336
```

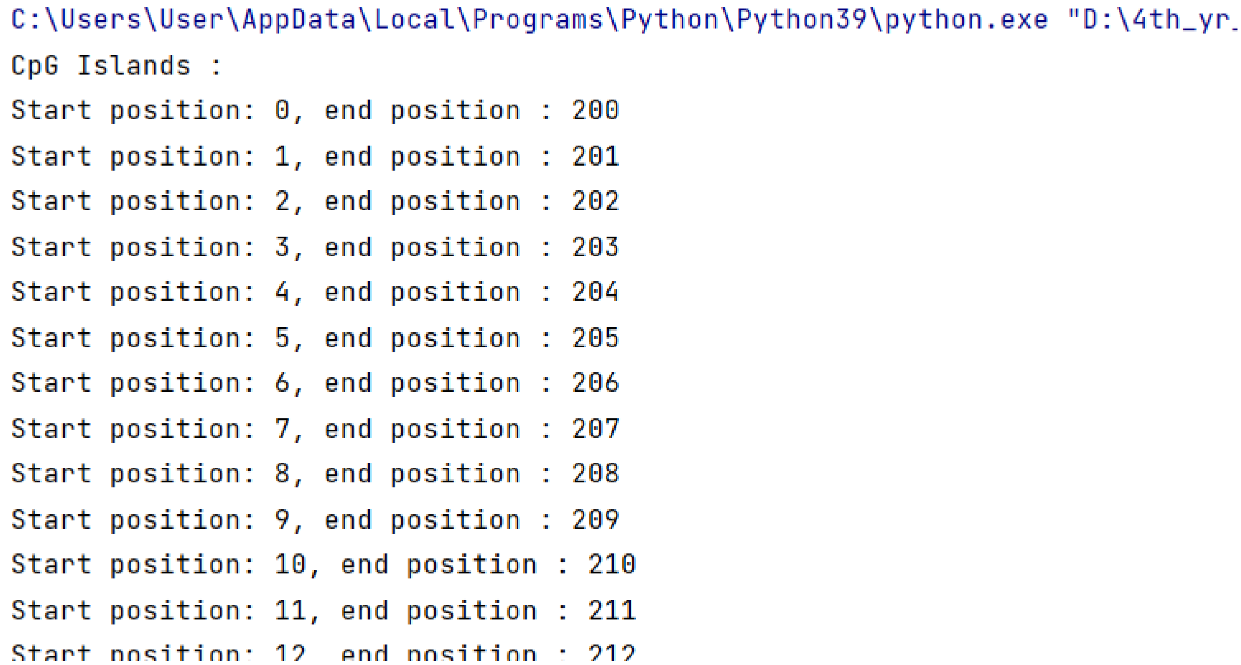


```
title: STRIDE: Systematic Radar Intelligence Analysis for ADRD Risk Evaluation with Gait Signature Simulation and Deep Learning.
authors: ['Cai F', 'Patharkar A', 'Wu T', 'Lure FYM', 'Chen H', 'Chen VC']
title: APOE loss-of-function variants: Compatible with longevity and associated with resistance to Alzheimer's Disease pathology.
authors: ['Chemparathy A', 'Guen YL', 'Chen S', 'Lee EG', 'Leong L', 'Gorzynski J', 'Xu G', 'Belloy M', 'Kasireddy N', 'Tauber AP', 'Williams K', 'Stewart I', 'Wingo T', 'Lah J',
title: Characterization of covalent inhibitors that disrupt the interaction between the tandem SH2 domains of SYK and FCER1G phospho-ITAM.
authors: ['Bashore FM', 'Katis VL', 'Du Y', 'Sikdar A', 'Wang D', 'Bradshaw WJ', 'Rygiel KA', 'Leisner TM', 'Chalk R', 'Mishra S', 'Williams AC', 'Gileadi O', 'Brennan PE', 'Wile
title: Full-Spectrum Neuronal Diversity and Stereotypy through Whole Brain Morphometry.
authors: ['Liu Y', 'Jiang S', 'Li Y', 'Zhao S', 'Yun Z', 'Zhao ZH', 'Zhang L', 'Wang G', 'Chen X', 'Manubens-Gil L', 'Hang Y', 'Garcia-Forn M', 'Wang W', 'Rubeis S', 'Wu Z', 'Ost
title: Spatial and single-nucleus transcriptomic analysis of genetic and sporadic forms of Alzheimer's Disease.
authors: ['Miyoshi E', 'Morabito S', 'Henningfield CM', 'Rahimzadeh N', 'Kiani Shabestari S', 'Das S', 'Michael N', 'Reese F', 'Shi Z', 'Cao Z', 'Scarfone V', 'Arreola MA', 'Lu J
title: Sexual coordination in a whole-brain map of prairie vole pair bonding.
authors: ['Gustison ML', 'Munoz-Castaneda R', 'Osten P', 'Phelps SM']
title: Image Analysis Techniques for In Vivo Quantification of Cerebrospinal Fluid Flow.
authors: ['Kim D', 'Gan Y', 'Nedergaard M', 'Kelley DH', 'Tithof J']
title: APOE4/4 is linked to damaging lipid droplets in Alzheimer's microglia.
authors: ['Haney MS', 'Palovics R', 'Munson CN', 'Long C', 'Johansson P', 'Yip O', 'Dong W', 'Rawat E', 'West E', 'Schlachetzki JC', 'Tsai A', 'Guldner IH', 'Lamichhane BS', 'Smi
title: Cuprizone drives divergent neuropathological changes in different mouse models of Alzheimer's disease.
authors: ['Cheng GW', 'Ma IW', 'Huang J', 'Yeung SH', 'Ho P', 'Chen Z', 'Mak HKF', 'Herrup K', 'Chan KWY', 'Tse KH']
title: Enhanced microglial dynamics and paucity of tau seeding in the amyloid plaque microenvironment contributes to cognitive resilience in Alzheimer's disease.
authors: ['Jury-Garfe N', 'You Y', 'Martinez P', 'Redding-Ochoa J', 'Karahan H', 'Johnson TS', 'Zhang J', 'Kim J', 'Troncoso JC', 'Lasagna-Reeves CA']
title: Looking at the Full Picture: Utilizing Topic Modeling to Determine Disease-Associated Microbiome Communities.
authors: ['Shrode RL', 'Ollberding NJ', 'Mangalam AK']
title: Comparative brain metabolomics reveals shared and distinct metabolic alterations in Alzheimer's disease and progressive supranuclear palsy.
authors: ['Batra R', 'Krumsiek J', 'Wang X', 'Allen M', 'Blach C', 'Kastenmuller G', 'Arnold M', 'Ertekin-Taner N', 'Kaddurah-Daouk RF']
title: A Natural Language Processing Algorithm for Classifying Suicidal Behaviors in Alzheimer's Disease and Related Dementia Patients: Development and Validation Using Electroni
authors: ['Zandbiglari K', 'Hasanzadeh HR', 'Kotecha P', 'Sajdeya R', 'Goodin AJ', 'Jiao T', 'Adiba FI', 'Mardini MT', 'Bian J', 'Rouhizadeh M']
title: Enterobacter hormaechei-Driven Novel Biosynthesis of Tin Oxide Nanoparticles and Evaluation of Their Anti-aging, Cytotoxic, and Enzyme Inhibition Potential.
```

3. Write a Biopython-program that finds CpG-islands from a given DNA-sequence.

```
# '''
# find CpG islands
# 08/12/2023
# Nimna Gamage
# s14682
# Lab 01-Question3_Sub_question3
# '''

#import modules
from Bio.SeqUtils import nt_search

#define function
def find_cpg_islands(dnaSequence, min_cpg_length=200, cpg_obs_exp_ratio=0.5):

    cpg_islands = []
    window_size = min_cpg_length

    for i in range(len(dnaSequence) - window_size + 1):
        sub = dnaSequence[i:i + window_size].upper()
        count_CG = sub.count('G') + sub.count('C')
        content_of_CG = count_CG / window_size
        obs = nt_search(sub, 'CG')

        if content_of_CG >= cpg_obs_exp_ratio and obs:
            cpg_islands.append((i, i + window_size))

    return cpg_islands


# given dna sequence
given_sequence =
"CGCGCGCGCGCGCCGGCGCGCGCGCGCGCGCGCATATATATATAGATAGATAGTAGCGCGCGCGCGCCGGCG
CGCGCGCGCGCGCGCGCGGCGCGCGCGCGCGCCGGCGCGCGCGCGCGCGCGCGGCGCGCGCGCGCGCGCGCGG
CGCGCGCGCGCGCGCGCGCGATCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGATCGCGCGCGCGCGC
GCGCGCGCGCGCGCGCGCGCGCGCG"

islands_cpg = find_cpg_islands(given_sequence)

#print each cpg island
print("CpG Islands : ")
for start, end in islands_cpg:
    print(f"Start position: {start}, end position : {end}")
```

Output;



```
C:\Users\User\AppData\Local\Programs\Python\Python39\python.exe "D:\4th_yr_
CpG Islands :
Start position: 0, end position : 200
Start position: 1, end position : 201
Start position: 2, end position : 202
Start position: 3, end position : 203
Start position: 4, end position : 204
Start position: 5, end position : 205
Start position: 6, end position : 206
Start position: 7, end position : 207
Start position: 8, end position : 208
Start position: 9, end position : 209
Start position: 10, end position : 210
Start position: 11, end position : 211
Start position: 12, end position : 212
```