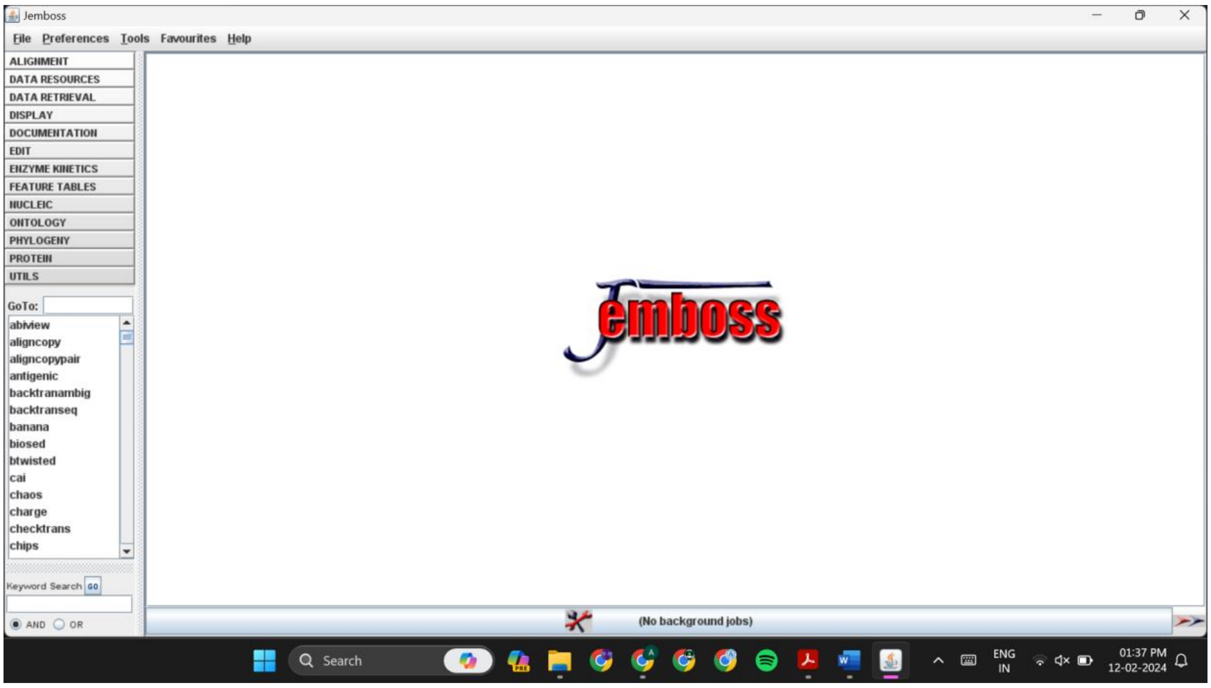
Anirudh Rao
BE21B004

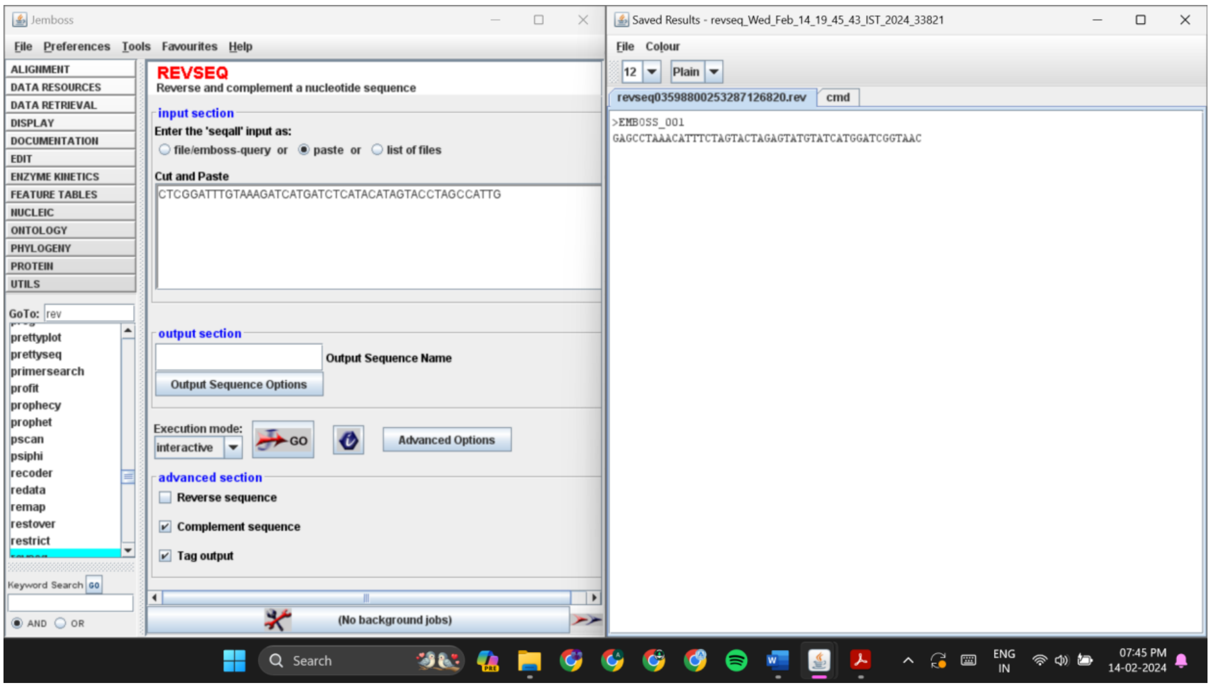
BT3040 – Bioinformatics

Practical 1

1.



2.



REVSEQ
Reverse and complement a nucleotide sequence

input section
Enter the 'seqall' input as: file/emboss-query or [✓ paste] or list of files

Cut and Paste
```
CTCGGATTTGTAAAGATCATGATCTCATACATAGTACCTAGCCATTG
```

output section
Output Sequence Name

Execution mode: interactive

advanced section
Reverse sequence
[✓] Complement sequence
[✓] Tag output

Saved Results - revseq_Wed_Feb_14_19_45_43_IST_2024_33821

revseq03598800253287126820.rev | cmd

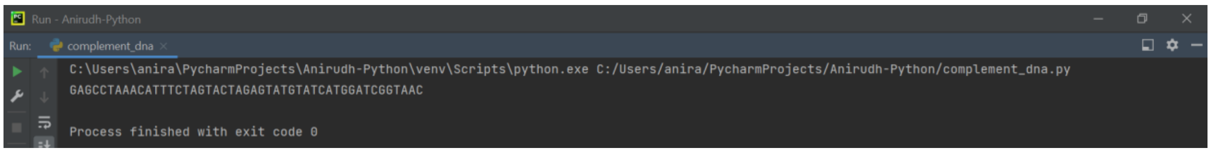
```
>EMBOSS_001
GAGCCTAAACATTTCTAGTACTAGAGTATGTATCATGGATCGGTAAC
```

3.

The Python code to find the complementary strand for the DNA sequence given in Q2 can be found below:

```python
def complement(sequence):

    """
    Returns the complementary sequence (in 3' to 5' direction) of a given
DNA sequence (in 5' to 3' direction)

    """

    complement = {"A": "T", "T": "A", "G": "C", "C": "G"}
    complementary = ""
    for base in sequence:
        complementary += complement[base]
    return complementary


test_seq = "CTCGGATTTGTAAAGATCATGATCTCATACATAGTACCTAGCCATTG"
ans = complement(test_seq)
print(ans)
```

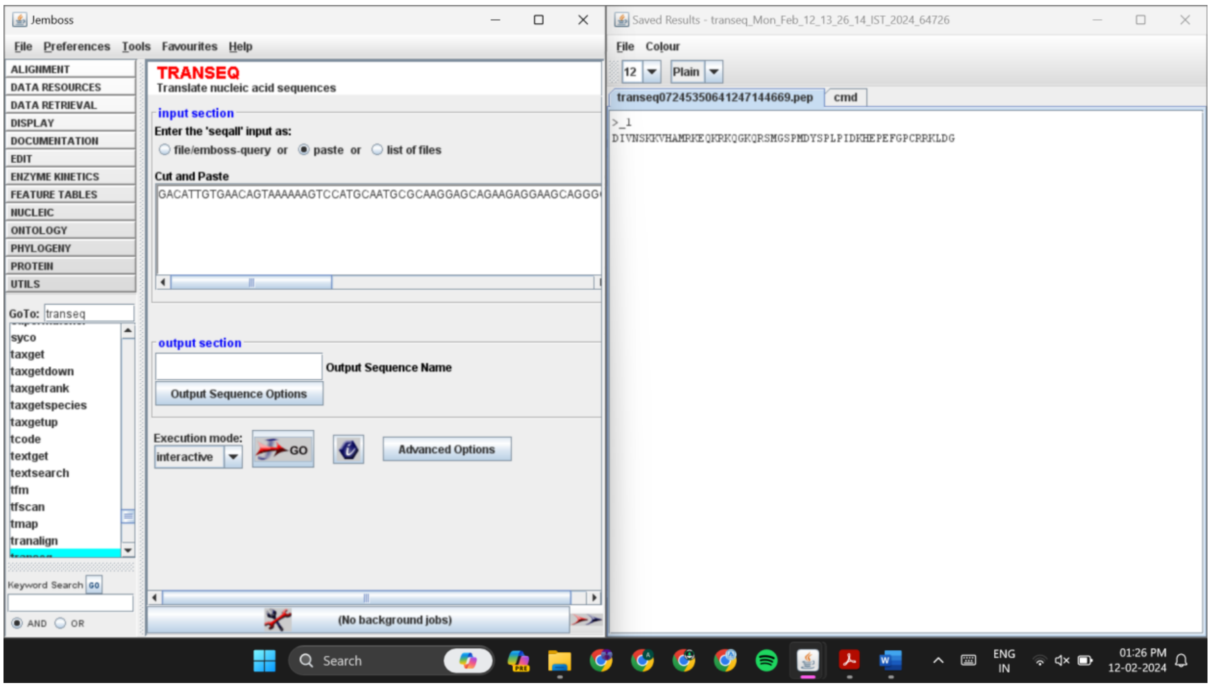
The output looks like:



```
C:\Users\anira\PycharmProjects\Anirudh-Python\venv\Scripts\python.exe C:/Users/anira/PycharmProjects/Anirudh-Python/complement_dna.py
GAGCCTAAACATTTCTAGTACTAGAGTATGTATCATGGATCGGTAAC

Process finished with exit code 0
```

Please note that the output is in the 3' to 5' direction.

4. (i)



TRANSEQ
Translate nucleic acid sequences

input section
Enter the 'seqall' input as: file/emboss-query or [✓ paste] or list of files

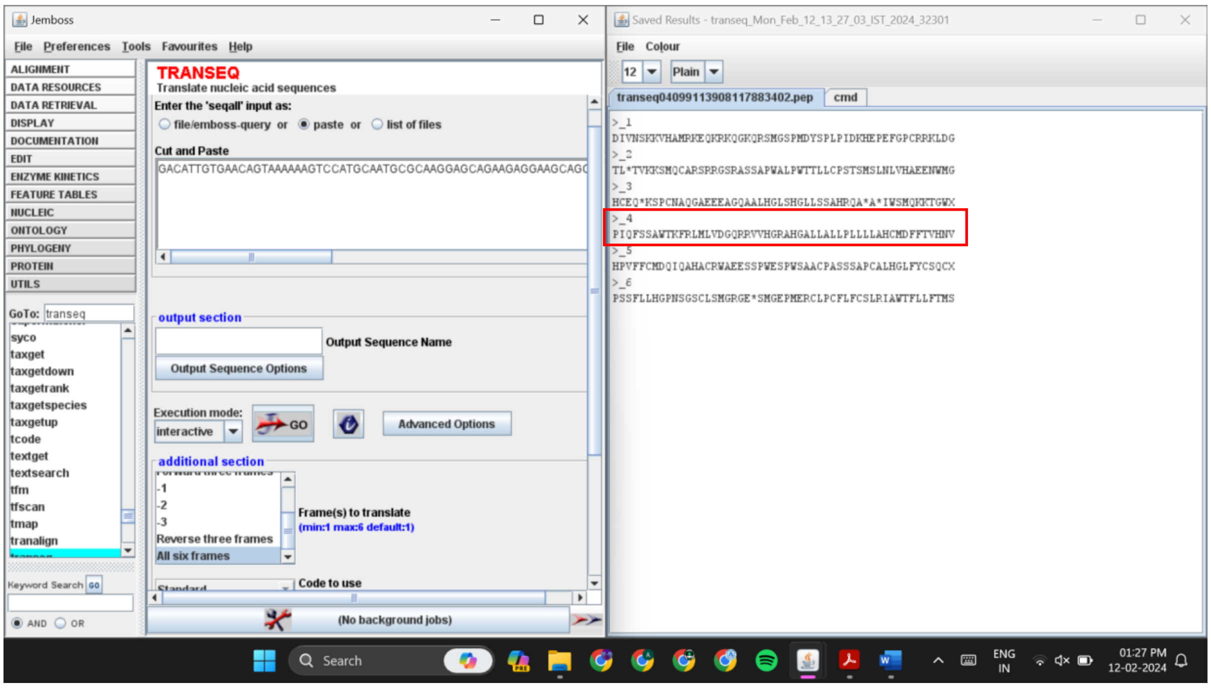
Cut and Paste
GACATTGTGAACAGTAAAAAAGTCCATGCAATGCGCAAGGAGCAGAAGAGGAAGCAGGG

Saved Results - transeq_Mon_Feb_12_13_26_14_IST_2024_64726

transeq07245350641247144669.pep

```
>_1
DIVNSKKVHAMRKEQKRKQGKQRSMGSPMDYSPLPIDKHEPEFGPCRRKLDG
```

4. (ii)



Enter the 'seqall' input as: ○ file/emboss-query or ● paste or ○ list of files

Cut and Paste: GACATTGTGAACAGTAAAAAAGTCCATGCAATGCGCAAGGAGCAGAAGAGGAAGCAGC

Execution mode: interactive

Frame(s) to translate (min:1 max:6 default:1): All six frames

Code to use: Standard

transeq0409911390811788340 2.pep

```
>_1
DIVNSKKVHAMRKEQKRKQGKQRSMGSPMDYSPLPIDKHEPEFGPCRRKLDG
>_2
TL*TVKKSMQCARSRRGSRASSAPWALPWTTLLCPSTSMSLNLVHAEENWMG
>_3
HCEQ*KSPCNAQGAEEEAGQAALHGLSHGLLSSAHRQA*A*IWSMQKKTGWX
>_4
PIQFSSAWTKFRLMLVDGQRRVVHGRAHGALLALLPLLLLAHCMDFFTVHNV
>_5
HPVFFCMDQIQAHACRWAEESSPWESPWSAACPASSSAPCALHGLFYCSQCX
>_6
PSSFLLHGPNSGSCLSMGRGE*SMGEPMERCLPCFLFCSLRIAWTFLLFTMS
```

The required reading frame is **reading frame 4**, which is the same as the **reading frame –1**.

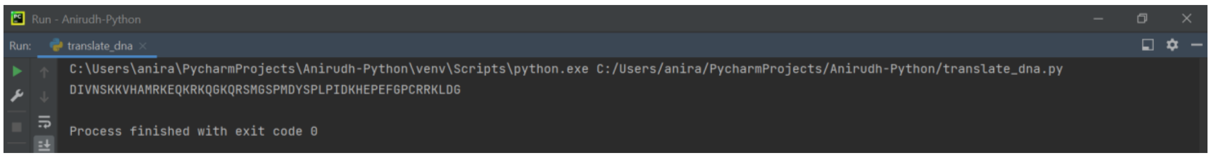
5.

The Python code to find the translated protein sequence for the DNA sequence given in Q4 can be found below:

```
def translate(sequence):

    """
    Returns the translated protein sequence of a given DNA sequence (in 5'
to 3' direction)

    """

    codon_table = {
    "GCT": "A", "GCC": "A", "GCA": "A", "GCG": "A",
    "TGT": "C", "TGC": "C",
    "GAT": "D", "GAC": "D",
    "GAA": "E", "GAG": "E",
    "TTT": "F", "TTC": "F",
    "GGT": "G", "GGC": "G", "GGA": "G", "GGG": "G",
    "CAT": "H", "CAC": "H",
    "ATA": "I", "ATT": "I", "ATC": "I",
    "AAA": "K", "AAG": "K",
    "TTA": "L", "TTG": "L", "CTT": "L", "CTC": "L", "CTA": "L", "CTG": "L",
    "ATG": "M",
    "AAT": "N", "AAC": "N",
    "CCT": "P", "CCC": "P", "CCA": "P", "CCG": "P",
    "CAA": "Q", "CAG": "Q",
    "CGT": "R", "CGC": "R", "CGA": "R", "CGG": "R", "AGA": "R", "AGG": "R",
    "TCT": "S", "TCC": "S", "TCA": "S", "TCG": "S", "AGT": "S", "AGC": "S",
    "ACT": "T", "ACC": "T", "ACA": "T", "ACG": "T",
```

```python
    "GTT": "V", "GTC": "V", "GTA": "V", "GTG": "V",
    "TGG": "W",
    "TAT": "Y", "TAC": "Y",
    "TAA": "*", "TAG": "*", "TGA": "*"}

    protein = ""
    for i in range(0, len(sequence) - 3 + 1, 3):
        codon = sequence[i:i+3]
        protein += codon_table[codon]

    return protein


test_seq =
"GACATTGTGAACAGTAAAAAAGTCCATGCAATGCGCAAGGAGCAGAAGAGGAAGCAGGGCAAGCAGCGCTCCAT
GGGCTCTCCCATGGACTACTCTCCTCTGCCCATCGACAAGCATGAGCCTGAATTTGGTCCATGCAGAAGAAAACT
GGATGGG"
ans = translate(test_seq)
print(ans)
```

The output looks like:



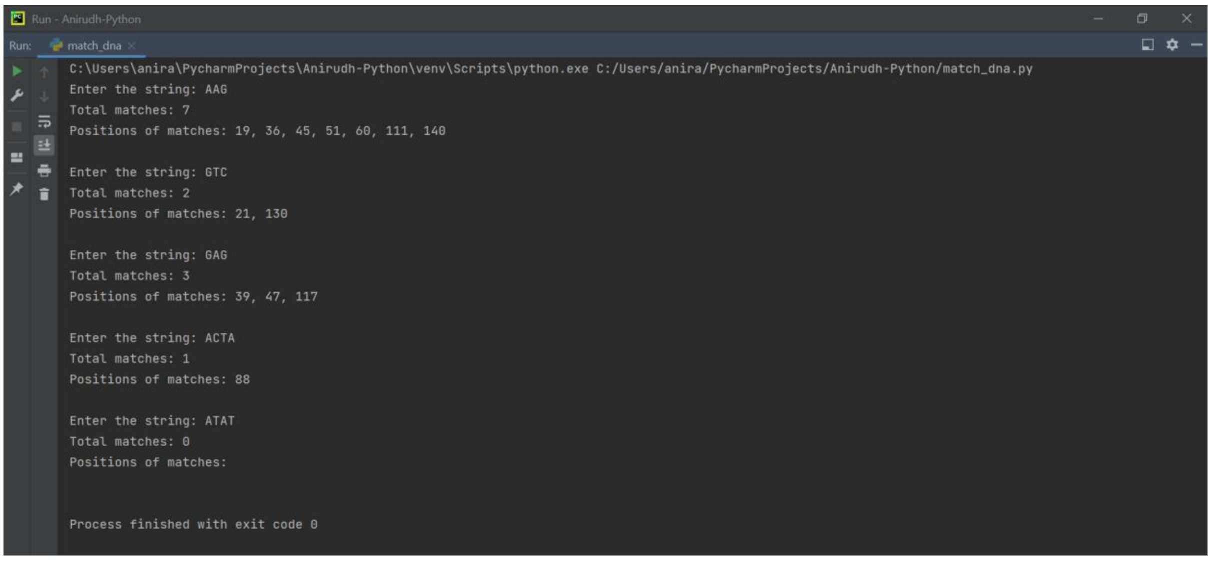
```
C:\Users\anira\PycharmProjects\Anirudh-Python\venv\Scripts\python.exe C:/Users/anira/PycharmProjects/Anirudh-Python/translate_dna.py
DIVNSKKVHAMRKEQKRKQGKQRSMGSPMDYSPLPIDKHEPEFGPCRRKLDG

Process finished with exit code 0
```

6.

The Python code to find the strings ‘AAG’, ‘GTC’, ‘GAG, ‘ACTA’, and ‘ATAT’ in the DNA sequence given in Q4 can be found below:

```python
def find_match(sequence, string):

    match_count = 0
    match_positions = []
    for i in range(len(sequence) - len(string) + 1):
        if sequence[i:i+len(string)] == string:
            match_count += 1
            match_positions.append(str(i))
    positions = ", ".join(match_positions)
    print(f"Enter the string: {string}")
    print(f"Total matches: {match_count}")
    print(f"Positions of matches: {positions}\n")


test_seq =
"GACATTGTGAACAGTAAAAAAGTCCATGCAATGCGCAAGGAGCAGAAGAGGAAGCAGGGCAAGCAGCGCTCCAT
GGGCTCTCCCATGGACTACTCTCCTCTGCCCATCGACAAGCATGAGCCTGAATTTGGTCCATGCAGAAGAAAACT
GGATGGG"
test_strings = ["AAG", "GTC", "GAG", "ACTA", "ATAT"]
for test_string in test_strings:
    find_match(test_seq, test_string)
```

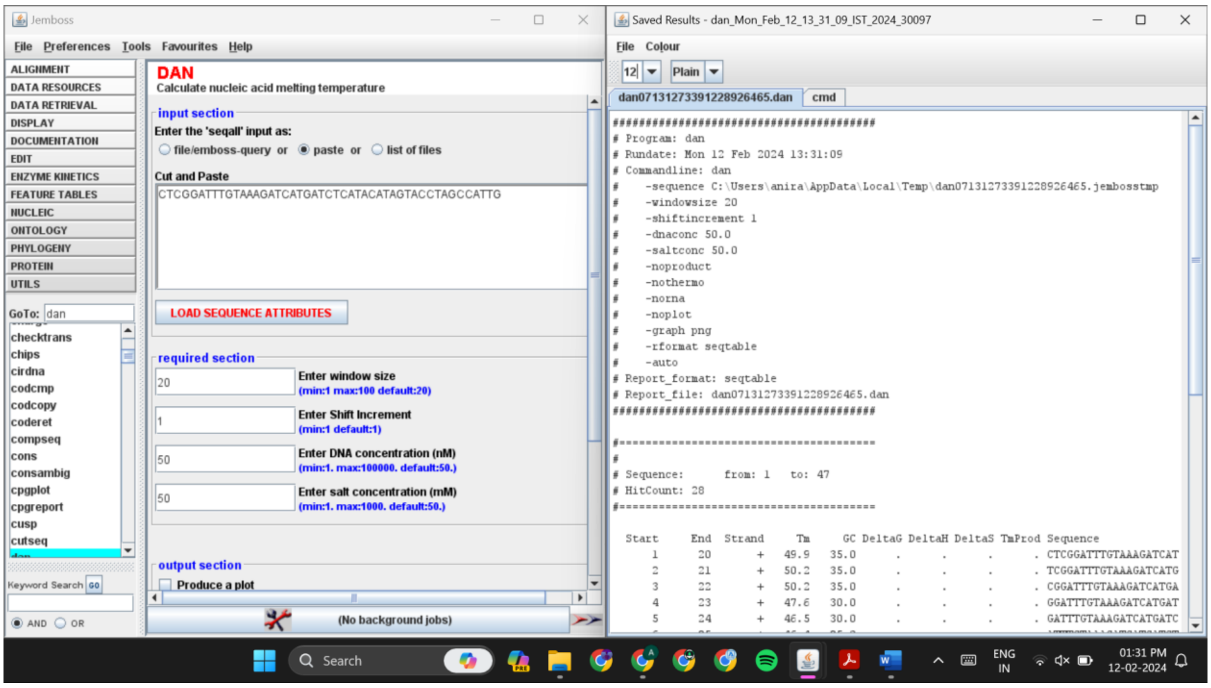
The output looks like:



```
C:\Users\anira\PycharmProjects\Anirudh-Python\venv\Scripts\python.exe C:/Users/anira/PycharmProjects/Anirudh-Python/match_dna.py
Enter the string: AAG
Total matches: 7
Positions of matches: 19, 36, 45, 51, 60, 111, 140

Enter the string: GTC
Total matches: 2
Positions of matches: 21, 130

Enter the string: GAG
Total matches: 3
Positions of matches: 39, 47, 117

Enter the string: ACTA
Total matches: 1
Positions of matches: 88

Enter the string: ATAT
Total matches: 0
Positions of matches:


Process finished with exit code 0
```

7.

DAN is used to calculate melting temperature while BANANA is used to plot the bending and curvature of DNA.



DAN
Calculate nucleic acid melting temperature

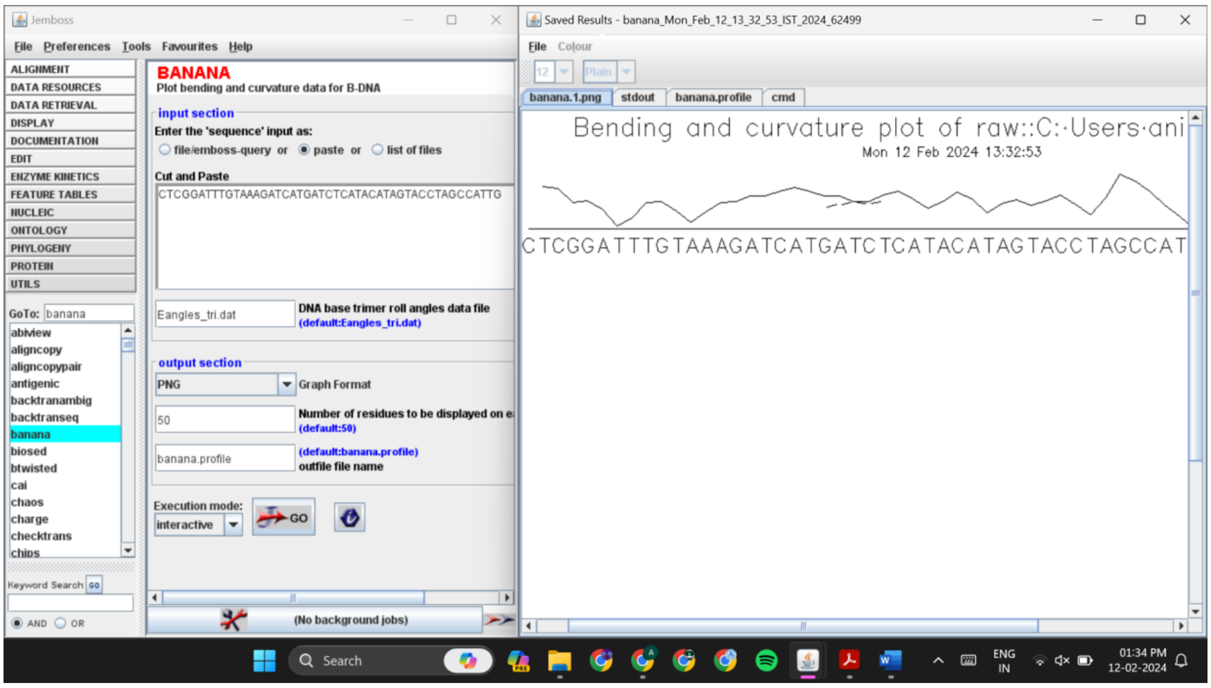
input section
Enter the 'seqall' input as: ○ file/emboss-query or [● paste] or ○ list of files
Cut and Paste
CTCGGATTTGTAAAGATCATGATCTCATACATAGTACCTAGCCATTG

required section
20 Enter window size (min:1 max:100 default:20)
1 Enter Shift Increment (min:1 default:1)
50 Enter DNA concentration (nM) (min:1. max:100000. default:50.)
50 Enter salt concentration (mM) (min:1. max:1000. default:50.)

output section
☐ Produce a plot

Saved Results - dan_Mon_Feb_12_13_31_09_IST_2024_30097
File Colour
12 Plain
dan07131273391228926465.dan | cmd

```
########################################
# Program: dan
# Rundate: Mon 12 Feb 2024 13:31:09
# Commandline: dan
#    -sequence C:\Users\anira\AppData\Local\Temp\dan07131273391228926465.jembosstmp
#    -windowsize 20
#    -shiftincrement 1
#    -dnaconc 50.0
#    -saltconc 50.0
#    -noproduct
#    -nothermo
#    -norna
#    -noplot
#    -graph png
#    -rformat seqtable
#    -auto
# Report_format: seqtable
# Report_file: dan07131273391228926465.dan
########################################

#=======================================
#
# Sequence:      from: 1   to: 47
# HitCount: 28
#=======================================

  Start     End  Strand      Tm      GC DeltaG DeltaH DeltaS TmProd Sequence
      1      20       +    49.9    35.0      .      .      .      . CTCGGATTTGTAAAGATCAT
      2      21       +    50.2    35.0      .      .      .      . TCGGATTTGTAAAGATCATG
      3      22       +    50.2    35.0      .      .      .      . CGGATTTGTAAAGATCATGA
      4      23       +    47.6    30.0      .      .      .      . GGATTTGTAAAGATCATGAT
      5      24       +    46.5    30.0      .      .      .      . GATTTGTAAAGATCATGATC
```



BANANA
Plot bending and curvature data for B-DNA

input section
Enter the 'sequence' input as: file/emboss-query or [✓ paste] or list of files

Cut and Paste
CTCGGATTTGTAAAGATCATGATCTCATACATAGTACCTAGCCATTG

Eangles_tri.dat — DNA base trimer roll angles data file (default:Eangles_tri.dat)

output section
PNG — Graph Format
50 — Number of residues to be displayed on e (default:50)
banana.profile — (default:banana.profile) outfile file name

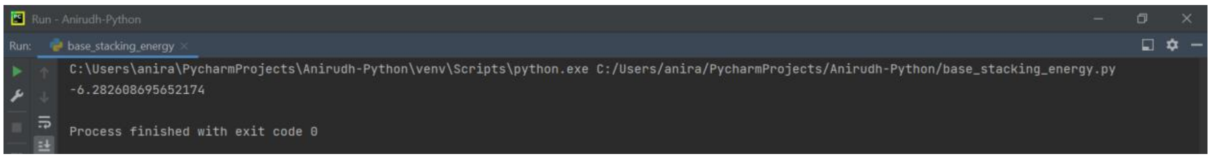
Execution mode: interactive

8.

The Python code to compute the average base stacking energy for the DNA sequence given in Q2 can be found below:

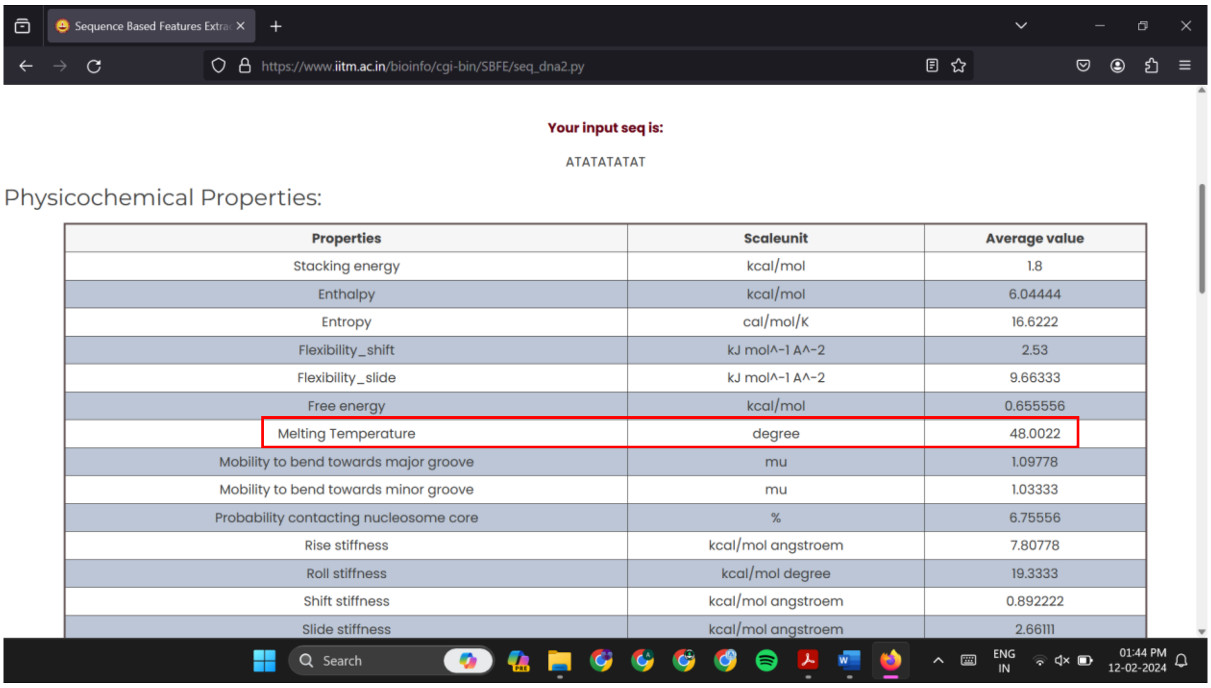
```python
def find_base_stack_energy(sequence):

    energies = {"AA": -4, "AT": -7, "AC": -5, "AG": -11, "TA": -7, "TT": -2, "TC": -3, "TG": -4, "CA": -9, "CT": -5, "CC": -6, "CG": -7, "GA": -9, "GT": -6, "GC": -4, "GG": -11}
    total_energy = 0
    for i in range(len(sequence)-2+1):
        dinucleotide = sequence[i:i+2]
        total_energy += energies[dinucleotide]
    avg_base_stack_energy = total_energy/(len(sequence)-2+1)
    return avg_base_stack_energy


test_seq = "CTCGGATTTGTAAAGATCATGATCTCATACATAGTACCTAGCCATTG"
print(find_base_stack_energy(test_seq))
```

The output looks like:



```
C:\Users\anira\PycharmProjects\Anirudh-Python\venv\Scripts\python.exe C:/Users/anira/PycharmProjects/Anirudh-Python/base_stacking_energy.py
-6.282608695652174

Process finished with exit code 0
```

9. (i)


Sequence Based Features Extra
https://www.iitm.ac.in/bioinfo/cgi-bin/SBFE/seq_dna2.py
Search
01:44 PM
12-02-2024

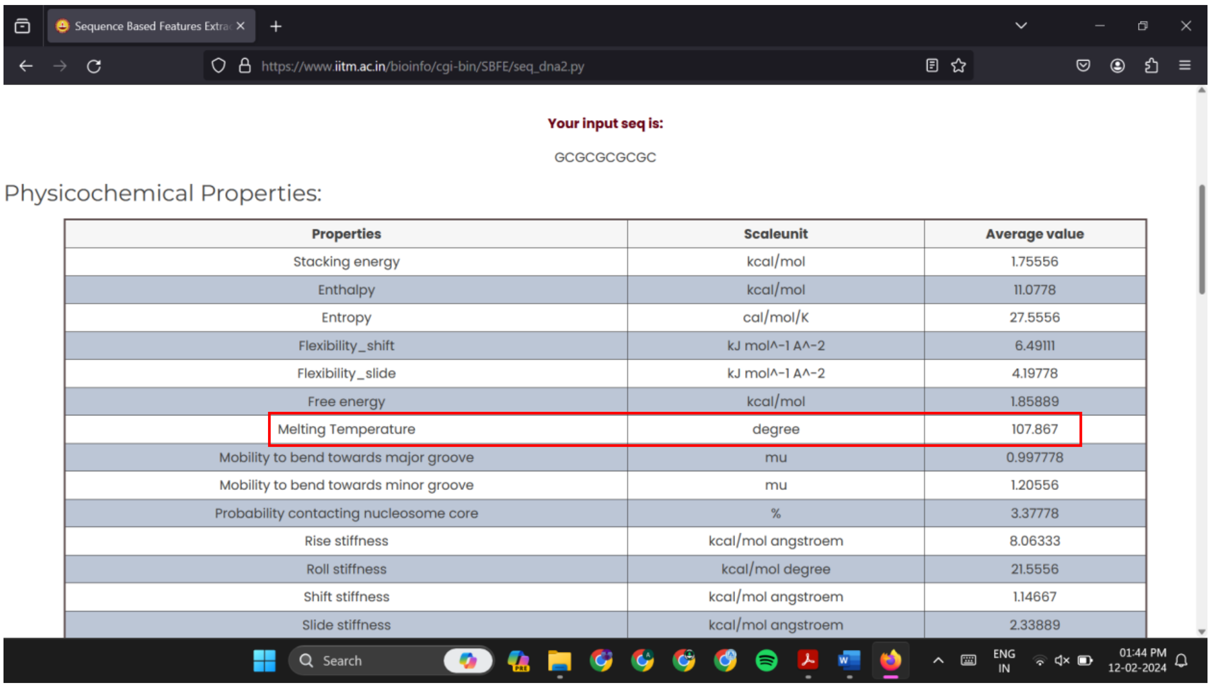
Your input seq is:

ATATATATAT

Physicochemical Properties:

Properties	Scaleunit	Average value
Stacking energy	kcal/mol	1.8
Enthalpy	kcal/mol	6.04444
Entropy	cal/mol/K	16.6222
Flexibility_shift	kJ mol^-1 A^-2	2.53
Flexibility_slide	kJ mol^-1 A^-2	9.66333
Free energy	kcal/mol	0.655556
Melting Temperature	degree	48.0022
Mobility to bend towards major groove	mu	1.09778
Mobility to bend towards minor groove	mu	1.03333
Probability contacting nucleosome core	%	6.75556
Rise stiffness	kcal/mol angstroem	7.80778
Roll stiffness	kcal/mol degree	19.3333
Shift stiffness	kcal/mol angstroem	0.892222
Slide stiffness	kcal/mol angstroem	2.66111

9. (ii)


Sequence Based Features Extra
https://www.iitm.ac.in/bioinfo/cgi-bin/SBFE/seq_dna2.py
Search
01:44 PM
12-02-2024

Your input seq is:

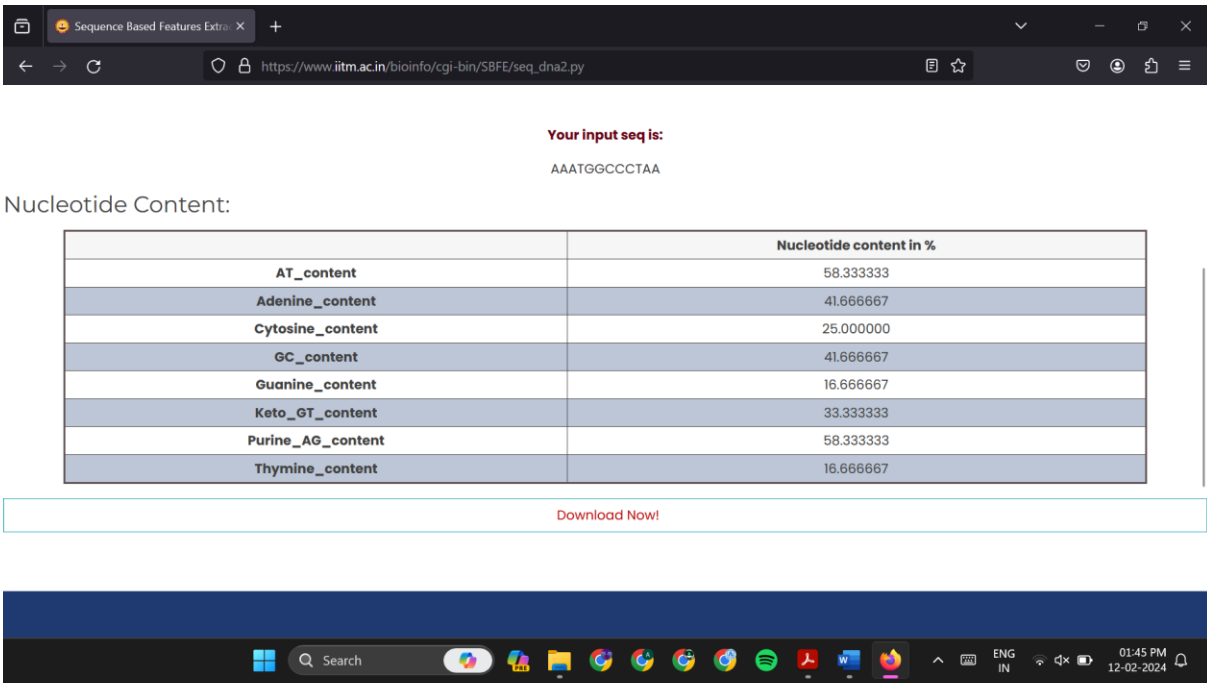
GCGCGCGCGC

Physicochemical Properties:

Properties	Scaleunit	Average value
Stacking energy	kcal/mol	1.75556
Enthalpy	kcal/mol	11.0778
Entropy	cal/mol/K	27.5556
Flexibility_shift	kJ mol^-1 A^-2	6.49111
Flexibility_slide	kJ mol^-1 A^-2	4.19778
Free energy	kcal/mol	1.85889
Melting Temperature	degree	107.867
Mobility to bend towards major groove	mu	0.997778
Mobility to bend towards minor groove	mu	1.20556
Probability contacting nucleosome core	%	3.37778
Rise stiffness	kcal/mol angstroem	8.06333
Roll stiffness	kcal/mol degree	21.5556
Shift stiffness	kcal/mol angstroem	1.14667
Slide stiffness	kcal/mol angstroem	2.33889

The melting temperature of the 1st sequence is much lower compared to the 2nd sequence. This is because the 1st sequence is AT rich while the 2nd sequence is GC rich. GC rich sequences will have a higher melting temperature due to the greater energy required to break the three hydrogen bonds between G and C rather than the two hydrogen bonds between A and T.

10.



Sequence Based Features Extra
https://www.iitm.ac.in/bioinfo/cgi-bin/SBFE/seq_dna2.py

Your input seq is:

AAATGGCCCTAA

Nucleotide Content:

	Nucleotide content in %
AT_content	58.333333
Adenine_content	41.666667
Cytosine_content	25.000000
GC_content	41.666667
Guanine_content	16.666667
Keto_GT_content	33.333333
Purine_AG_content	58.333333
Thymine_content	16.666667

Download Now!
Search
ENG
IN
01:45 PM
12-02-2024