BT3040 – BIOINFORMATICS – Assignment 6

Submitted by Sahana (BE17B038)

Question 1 –

Code –

```
import numpy as np

def composition(seq):
    AA_all
=['A','C','D','E','F','G','H','I','K','L','M','N','P','Q','R','S','T','V','W','Y'
]
    n = len(seq)
    composition = [0]*20
    for I in range(n):
        aa = seq[i]
        ind = AA_all.index(aa)
        composition[ind]+=1
    composition = np.divide(composition,(n/100))
    print('AA composition for the given sequence - ')
    for I in range(20):
        print('%s - %4.3f' %(AA_all[i],composition[i]))
seq1 = 'RATPTRWPVGCFNRPWTKWSYDEALDGIKAAGYAWTGLLTASKPSLHHATATPEYLAALKQKSRHAA'
seq2 = 'AAAVMMGLAAIGAAIGIGILGGKFLEGAARQPDLIPLLRTQFFIVMGLVDAIPMIAVGLGLYVMFAVA'
seq3 =
'QSGMTYRLGLSWDWDKSWWQTSTGRLTGYWDAGYTYWEGGDEGAGKHSLSFAPVFVYEFAGDSIKPFIEAGIGVAAFSGT
RVGDQNLGSSLNFEDRIGAGLKFANGQSVGVRAIHYSNAGLKQPNDGIESYSLFYKIPI'
composition(seq1)
composition(seq2)
composition(seq3)
```

Amino acids	Sequence 1	Sequence 2	Sequence 3
A	17.91	19.118	7.914
C	1.493	0	0
D	2.985	2.941	5.755
E	2.985	1.471	4.317
F	1.493	5.882	5.755
G	5.97	14.706	15.108
H	4.478	0	1.439
I	1.493	11.765	5.755
K	7.463	1.471	4.317
L	8.955	13.235	6.475
M	0	7.353	0.719
N	1.493	0	3.597
P	7.463	4.412	2.878
Q	1.493	2.941	3.597
R	5.97	2.941	3.597
S	5.97	0	10.072
T	10.448	1.471	4.317

V	1.493	8.824	4.317
W	5.97	0	4.317
Y	4.478	1.471	5.755

Analyses –

- Sequence 1 has a higher percentage of Alanine, Threonine, Leucine, Lysine and Proline. This means that the sequence is highly hydrophobic and slightly hydrophilic (due to presence of Threonine). Therefore, this polypeptide sequence must be found in the inner core of a protein in the usual hydrophilic environment where water is found in abundance, in the exterior space.
- Sequence 2 is extremely hydrophobic due to higher percentage of Alanine, Glycine, Isoleucine and Leucine. Therefore, this polypeptide sequence must be found in the inner core of a protein in the usual hydrophilic environment where water is found in abundance, in the exterior space.
- Sequence 3 has higher percentage of Glycine, Serine and Alanine. Serine makes this protein slightly hydrophilic which would otherwise be hydrophobic due to the presence of Gly and Ala.

Question 2 –

Algorithm –

- The program adds up all the molecular masses of each amino acid in the sequence.
- To account for the loss of water molecules during the formation of a peptide bond between any two adjacent amino acids, for a polypeptide sequence of length N, (N-1)*(Molecular weight of water) is subtracted.

Code –

```
def mol_weight(seq):
    MM= {'A': 85, 'C': 115, 'D': 130, 'E': 145, 'F': 160,
         'G': 70, 'W': 200, 'H': 150, 'I': 125, 'K': 145,
         'L': 125, 'M': 143, 'N': 130, 'Y': 175, 'P': 110,
         'Q': 140, 'R': 170, 'S': 100, 'T': 115, 'V': 110}
    n = len(seq)
    weight = 0
    for i in range(n):
        weight+=MM[seq[i]]
    weight-=(n-1)*18 #Removing the weight of water molecules that are removed by
formation of peptide bond
    print('Molecular weight of sequence is = %5.2f units.' %weight)

seq1 = 'RATPTRWPVGCFNRPWTKWSYDEALDGIKAAGYAWTGLLTASKPSLHHATATPEYLAALKQKSRHAA'
seq2 = 'AAAVMMGLAAIGAAIGIGILGGKFLEGAARQPDLIPLLRTQFFIVMGLVDAIPMIAVGLGLYVMFAVA'
seq3 =
'QSGMTYRLGLSWDWDKSWWQTSTGRLTGYWDAGYTYWEGGDEGAGKHSLSFAPVFVYEFAGDSIKPFIEAGIGVAAFSGT
RVGDQNLGSSLNFEDRIGAGLKFANGQSVGVRAIHYSNAGLKQPNDGIESYSLFYKIPI'

mol_weight(seq1)
mol_weight(seq2)
mol_weight(seq3)
```

OUTPUT (in order)-

```
Molecular weight of sequence is = 7127.00 units.
Molecular weight of sequence is = 6529.00 units.
Molecular weight of sequence is = 14539.00 units.
```

Question 3 –

Algorithm –

- The composition of each amino acids in the given sequence is first calculated. (Function from question 1 is called).
- The absolute difference between the composition of sequence with compositions of AAs in group A and B are computed.
- Least difference from above calculation would mean higher chances that the given sequence is closely related to that group.

Code –

```
def classification(seq):
    group_A = {'A': 8.47, 'D': 5.97, 'C': 1.39, 'E': 6.32, 'T': 5.79,
               'F': 3.91, 'G': 7.82, 'H': 2.26, 'I': 5.71, 'V': 7.02,
               'K': 5.76, 'L': 8.48, 'M': 2.21, 'N': 4.54, 'W': 1.44,
               'P': 4.63, 'Q': 3.82, 'R': 4.93, 'S': 5.94, 'Y': 3.58}
    group_B = {'A': 8.95, 'D': 5.91, 'C': 0.47, 'E': 4.78, 'T': 6.54,
               'F': 3.68, 'G': 8.54, 'H': 1.25, 'I': 4.77, 'V': 6.76,
               'K': 4.93, 'L': 8.78, 'M': 1.56, 'N': 5.74, 'W': 1.24,
               'P': 3.74, 'Q': 4.75, 'R': 5.24, 'S': 8.05, 'Y': 4.13}

    AA_all
=['A','C','D','E','F','G','H','I','K','L','M','N','P','Q','R','S','T','V','W','Y'
]
    n = len(seq)
    comp = composition(seq)
    seq_A = 0
    seq_B = 0
    for i in range(n):
        aa = seq[i]
        ind = AA_all.index(aa)
        seq_A+= abs(comp[ind]-group_A[aa])
        seq_B+= abs(comp[ind]-group_B[aa])
    print('Deviation score of given sequence from Group A = %4.3f.' %seq_A)
    print('Deviation score of given sequence from Group B = %4.3f.' %seq_B)
    if seq_A < seq_B:
        print('\nGiven sequence appears to be close to Group A.\n')
    else:
        print('\nGiven sequence appears to be close to Group B.\n')

seq1 = 'RATPTRWPVGCFNRPWTKWSYDEALDGIKAAGYAWTGLLTASKPSLHHATATPEYLAALKQKSRHAA'
seq2 = 'AAAVMMGLAAIGAAIGIGILGGKFLEGAARQPDLIPLLRTQFFIVMGLVDAIPMIAVGLGLYVMFAVA'
seq3 =
'QSGMTYRLGLSWDWDKSWWQTSTGRLTGYWDAGYTYWEGGDEGAGKHSLSFAPVFVYEFAGDSIKPFIEAGIGVAAFSGT
RVGDQNLGSSLNFEDRIGAGLKFANGQSVGVRAIHYSNAGLKQPNDGIESYSLFYKIPI'

classification(seq1)
classification(seq2)
classification(seq3)
```

OUTPUT (in order) -

```
Deviation score of given sequence from Group A = 240.844.
Deviation score of given sequence from Group B = 247.073.

Given sequence appears to be close to Group A.

Deviation score of given sequence from Group A = 370.954.
Deviation score of given sequence from Group B = 370.715.

Given sequence appears to be close to Group B.

Deviation score of given sequence from Group A = 354.943.
Deviation score of given sequence from Group B = 319.389.

Given sequence appears to be close to Group B.
```

Question 4 –

CODE -

```
def pair_pref(seq):
    AA_all
=['A','C','D','E','F','G','H','I','K','L','M','N','P','Q','R','S','T','V','W','Y'
]
    n = len(seq)
    pref1 = [[0 for i in range(20)] for i in range(20)]
    pref2 = [[0 for i in range(20)] for i in range(20)]
    pref3 = [[0 for i in range(20)] for i in range(20)]

    comp = [0]*20
    for i in range(n):
        aa = seq[i]
        ind = AA_all.index(aa)
        comp[ind]+=1

    for i in range(20):
        for j in range(20):
            pair = AA_all[i]+AA_all[j]
            for k in range(n-1):
                if pair==seq[k:k+2]:
                    pref1[i][j]+=1
                    pref2[i][j]+=1
                    pref3[i][j]+=1
            if comp[i]==0 or comp[j]==0:
                pref1[i][j]=0
                pref2[i][j]=0
                pref3[i][j]=0
            else:
                pref1[i][j]=(pref1[i][j]*100)/(comp[i]+comp[j])
                pref2[i][j]=(pref2[i][j]*100)/(n-1)
                pref3[i][j]=(pref3[i][j]*100)/(comp[i]*comp[j])

    print('\nPair-wise preference score for the given sequnce-')
    print('\nFormula 1 : ')
    for i in range(20):
        for j in range(20):
```

```
            print('%4.2f' %pref1[i][j], end=' ')
        print('\n')

    print('\nFormula 2 : ')
    for i in range(20):
        for j in range(20):
            print('%4.2f' %pref2[i][j], end=' ')
        print('\n')

    print('\nFormula 3 : ')
    for i in range(20):
        for j in range(20):
            print('%4.2f' %pref3[i][j], end=' ')
        print('\n')

    return pref1,pref2,pref3

pair_pref(seq1)
pair_pref(seq2)
pair_pref(seq3)

OUTPUT –
```

Sequence 1 – Formula 1

	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
A	12.5	0	0	0	0	6.2	0	0	0	11.1	0	0	0	0	0	6.2	15.8	0	6.2	0
C	0	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	25	0	16.7	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	7.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20
F	0	0	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0
G	0	20	0	0	0	0	0	20	0	10	0	0	0	0	0	0	0	0	0	14.3
H	13.3	0	0	0	0	0	16.7	0	0	0	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	0	16.7	0	0	0	0	0	0	0	0	0	0	0
K	5.9	0	0	0	0	0	0	0	0	0	0	0	10	16.7	0	11.1	0	0	11.1	0
L	5.6	0	12.5	0	0	0	11.1	0	9.1	8.3	0	0	0	0	0	0	7.7	0	0	0
M	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	20	0	0	0	0	0
P	0	0	0	14.3	0	0	0	0	0	0	0	0	0	0	0	11.1	8.3	16.7	11.1	0
Q	0	0	0	0	0	0	0	0	16.7	0	0	0	0	0	0	0	0	0	0	0

R	6.2	0	0	0	0	0	14.3	0	0	0	0	0	11.1	0	0	0	0	0	12.5	0
S	0	0	0	0	0	0	0	0	11.1	10	0	0	0	0	12.5	0	0	0	0	14.3
T	10.5	0	0	0	0	9.1	0	0	8.3	0	0	0	16.7	0	9.1	0	0	0	0	0
V	0	0	0	0	0	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	11.1	0	0	12.5	18.2	0	0	0
Y	6.7	0	20	0	0	0	0	0	0	11.1	0	0	0	0	0	0	0	0	0	0

Sequence 1 – Formula 2

	A	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**K**	**L**	**M**	**N**	**P**	**Q**	**R**	**S**	**T**	**V**	**W**	**Y**
A	4.5	0	0	0	0	1.5	0	0	0	3	0	0	0	0	0	1.5	4.5	0	1.5	0
C	0	0	0	0	1.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	1.5	0	1.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	1.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.5
F	0	0	0	0	0	0	0	0	0	0	0	1.5	0	0	0	0	0	0	0	0
G	0	1.5	0	0	0	0	0	1.5	0	1.5	0	0	0	0	0	0	0	0	0	1.5
H	3	0	0	0	0	0	1.5	0	0	0	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	0	1.5	0	0	0	0	0	0	0	0	0	0	0
K	1.5	0	0	0	0	0	0	0	0	0	0	0	1.5	1.5	0	1.5	0	0	1.5	0
L	1.5	0	1.5	0	0	0	1.5	0	1.5	1.5	0	0	0	0	0	0	1.5	0	0	0
M	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.5	0	0	0	0	0
P	0	0	0	1.5	0	0	0	0	0	0	0	0	0	0	0	1.5	1.5	1.5	1.5	0
Q	0	0	0	0	0	0	0	0	1.5	0	0	0	0	0	0	0	0	0	0	0
R	1.5	0	0	0	0	0	1.5	0	0	0	0	0	1.5	0	0	0	0	0	1.5	0
S	0	0	0	0	0	0	0	0	1.5	1.5	0	0	0	0	1.5	0	0	0	0	1.5
T	3	0	0	0	0	1.5	0	0	1.5	0	0	0	3	0	1.5	0	0	0	0	0
V	0	0	0	0	0	1.	0	0	0	0	0	0	0	0	0	0	0	0	0	0

						5														
W	0	0	0	0	0	0	0	0	0	0	0	0	1.5	0	0	1.5	3	0	0	0
Y	1.5	0	1.5	0	0	0	0	0	0	1.5	0	0	0	0	0	0	0	0	0	0

Sequence 1 – Formula 3

	A	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**K**	**L**	**M**	**N**	**P**	**Q**	**R**	**S**	**T**	**V**	**W**	**Y**
A	2.1	0	0	0	0	2.1	0	0	0	2.8	0	0	0	0	0	2.1	3.6	0	2.1	0
C	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	25	0	12.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	4.2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16.7
F	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0
G	0	25	0	0	0	0	0	25	0	4.2	0	0	0	0	0	0	0	0	0	8.3
H	5.6	0	0	0	0	0	11.1	0	0	0	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	0	20	0	0	0	0	0	0	0	0	0	0	0
K	1.7	0	0	0	0	0	0	0	0	0	0	0	4	20	0	5	0	0	5	0
L	1.4	0	8.3	0	0	0	5.6	0	3.3	2.8	0	0	0	0	0	0	2.4	0	0	0
M	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25	0	0	0	0	0
P	0	0	0	10	0	0	0	0	0	0	0	0	0	0	0	5	2.9	20	5	0
Q	0	0	0	0	0	0	0	0	20	0	0	0	0	0	0	0	0	0	0	0
R	2.1	0	0	0	0	0	8.3	0	0	0	0	0	5	0	0	0	0	0	6.2	0
S	0	0	0	0	0	0	0	0	5	4.2	0	0	0	0	6.2	0	0	0	0	8.3
T	2.4	0	0	0	0	3.6	0	0	2.9	0	0	0	5.7	0	3.6	0	0	0	0	0
V	0	0	0	0	0	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	6.2	7.1	0	0	0
Y	2.8	0	16.7	0	0	0	0	0	0	5.6	0	0	0	0	0	0	0	0	0	0

Sequence 2 – Formula 1

	A	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**K**	**L**	**M**	**N**	**P**	**Q**	**R**	**S**	**T**	**V**	**W**	**Y**

A	19.2	0	0	0	0	0	0	14.3	0	0	0	0	0	0	6.7	0	0	15.8	0	0
C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	6.7	0	0	0	0	0	0	0	0	9.1	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	9.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
F	5.9	0	0	0	12.5	0	0	8.3	0	7.7	0	0	0	0	0	0	0	0	0	0
G	8.7	0	0	0	0	5	0	11.1	9.1	21.1	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I	4.8	0	0	0	0	16.7	0	0	0	5.9	0	0	18.2	0	0	0	0	7.1	0	0
K	0	0	0	0	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L	4.5	0	0	10	0	10.5	0	5.9	0	5.6	0	0	0	0	9.1	0	0	6.7	0	10
M	0	0	0	0	11.1	13.3	0	7.7	0	0	10	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P	0	0	20	0	0	0	0	0	0	8.3	12.5	0	0	0	0	0	0	0	0	0
Q	0	0	0	0	16.7	0	0	0	0	0	0	0	20	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	25	0	0	33.3	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
T	0	0	0	0	0	0	0	0	0	0	0	0	0	33.3	0	0	0	0	0	0
V	5.3	0	12.5	0	0	6.2	0	0	0	0	27.3	0	0	0	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	14.3	0	0

Sequence 2 – Formula 2

	A	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**K**	**L**	**M**	**N**	**P**	**Q**	**R**	**S**	**T**	**V**	**W**	**Y**
A	7.5	0	0	0	0	0	0	4.5	0	0	0	0	0	0	1.5	0	0	4.5	0	0
C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	1.5	0	0	0	0	0	0	0	0	1.5	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	1.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
F	1.5	0	0	0	1.5	0	0	1.5	0	1.5	0	0	0	0	0	0	0	0	0	0
G	3	0	0	0	0	1.5	0	3	1.5	6	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I	1.5	0	0	0	0	4.5	0	0	0	1.5	0	0	3	0	0	0	0	1.5	0	0
K	0	0	0	0	1.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L	1.5	0	0	1.5	0	3	0	1.5	0	1.5	0	0	0	0	1.5	0	0	1.5	0	1.5
M	0	0	0	0	1.5	3	0	1.5	0	0	1.5	0	0	0	0	0	0	0	0	0

N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P	0	0	1.5	0	0	0	0	0	0	1.5	1.5	0	0	0	0	0	0	0	0	0
Q	0	0	0	0	1.5	0	0	0	0	0	0	0	1.5	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	1.5	0	0	1.5	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
T	0	0	0	0	0	0	0	0	0	0	0	0	0	1.5	0	0	0	0	0	0
V	1.5	0	1.5	0	0	1.5	0	0	0	0	4.5	0	0	0	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.5	0	0

Sequence 2 – Formula 3

	A	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**K**	**L**	**M**	**N**	**P**	**Q**	**R**	**S**	**T**	**V**	**W**	**Y**
A	3	0	0	0	0	0	0	2.9	0	0	0	0	0	0	3.8	0	0	3.8	0	0
C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	3.8	0	0	0	0	0	0	0	0	5.6	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
F	1.9	0	0	0	6.2	0	0	3.1	0	2.8	0	0	0	0	0	0	0	0	0	0
G	1.5	0	0	0	0	1	0	2.5	10	4.4	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I	1	0	0	0	0	3.8	0	0	0	1.4	0	0	8.3	0	0	0	0	2.1	0	0
K	0	0	0	0	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L	0.9	0	0	11.1	0	2.2	0	1.4	0	1.2	0	0	0	0	5.6	0	0	1.9	0	11.1
M	0	0	0	0	5	4	0	2.5	0	0	4	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P	0	0	16.7	0	0	0	0	0	0	3.7	6.7	0	0	0	0	0	0	0	0	0
Q	0	0	0	0	12.5	0	0	0	0	0	0	0	16.7	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	25	0	0	50	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
T	0	0	0	0	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0
V	1.3	0	8.3	0	0	1.7	0	0	0	0	10	0	0	0	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16.7	0	0

Sequence 3 – Formula 1

	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
A	4.5	0	0	0	5.3	18.8	0	5.3	0	0	0	6.2	6.7	0	0	0	0	0	0	0
C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	5.3	0	0	7.1	0	3.4	0	0	7.1	0	0	0	0	7.7	7.7	4.5	0	0	7.1	0
E	5.9	0	7.1	0	7.1	7.4	0	0	0	0	0	0	0	0	0	5	0	0	0	0
F	15.8	0	0	7.1	0	0	0	6.2	0	0	0	0	0	0	0	4.5	0	7.1	0	6.2
G	6.2	0	10.3	0	0	2.4	0	6.9	3.7	10	4.5	0	0	3.8	3.8	2.9	3.7	7.4	0	6.9
H	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6.2	0	0	0	10
I	0	0	0	14.3	0	6.9	10	0	7.1	0	0	0	8.3	0	0	0	0	0	0	0
K	0	0	0	0	7.1	0	12.5	7.1	0	0	0	0	10	9.1	0	5	0	0	0	0
L	0	0	0	0	5.9	6.7	0	0	13.3	0	0	7.1	0	0	0	8.7	6.7	0	0	0
M	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	14.3	0	0	0
N	6.2	0	7.7	0	7.7	3.8	0	0	0	7.1	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	8.3	0	0	8.3	0	0	0	11.1	0	0	0	0	0	10	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	10	11.1	0	0	10.5	9.1	0	0	0
R	6.2	0	0	0	0	0	0	7.7	0	14.3	0	0	0	0	0	0	0	9.1	0	0
S	0	0	0	0	4.5	5.7	0	4.5	0	13	0	5.3	0	0	0	3.6	5	5	10	4.5
T	0	0	0	0	0	7.4	0	0	0	0	0	0	0	0	9.1	5	0	0	0	14.3
V	5.9	0	0	0	7.1	7.4	0	0	0	0	0	0	0	0	9.1	0	0	0	0	7.1
W	0	0	21.4	8.3	0	0	0	0	0	0	0	0	0	9.1	0	0	0	0	8.3	0
Y	0	0	0	7.1	0	0	0	0	7.1	0	0	0	0	0	7.7	9.1	7.1	0	14.3	0

Sequence 3 – Formula 2

	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
A	0.7	0	0	0	0.7	4.3	0	0.7	0	0	0	0.7	0.7	0	0	0	0	0	0	0
C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0.	0	0	0.	0	0.	0	0	0.	0	0	0	0	0.	0.	0.	0	0	0.	0

	7			7		7			7					7	7	7			7	
E	0.7	0	0.7	0	0.7	1.4	0	0	0	0	0	0	0	0	0	0.7	0	0	0	0
F	2.2	0	0	0.7	0	0	0	0.7	0	0	0	0	0	0	0	0.7	0	0.7	0	0.7
G	1.4	0	2.2	0	0	0.7	0	1.4	0.7	2.2	0.7	0	0	0.7	0.7	0.7	0.7	1.4	0	1.4
H	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.7	0	0	0	0.7
I	0	0	0	1.4	0	1.4	0.7	0	0.7	0	0	0	0.7	0	0	0	0	0	0	0
K	0	0	0	0	0.7	0	0.7	0.7	0	0	0	0	0.7	0.7	0	0.7	0	0	0	0
L	0	0	0	0	0.7	1.4	0	0	1.4	0	0	0.7	0	0	0	1.4	0.7	0	0	0
M	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.7	0	0	0
N	0.7	0	0.7	0	0.7	0.7	0	0	0	0.7	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0.7	0	0	0.7	0	0	0	0.7	0	0	0	0	0	0.7	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0.7	0.7	0	0	1.4	0.7	0	0	0
R	0.7	0	0	0	0	0	0	0.7	0	1.4	0	0	0	0	0	0	0	0.7	0	0
S	0	0	0	0	0.7	1.4	0	0.7	0	2.2	0	0.7	0	0	0	0.7	0.7	0.7	1.4	0.7
T	0	0	0	0	0	1.4	0	0	0	0	0	0	0	0	0.7	0.7	0	0	0	1.4
V	0.7	0	0	0	0.7	1.4	0	0	0	0	0	0	0	0	0.7	0	0	0	0	0.7
W	0	0	2.2	0.7	0	0	0	0	0	0	0	0	0	0.7	0	0	0	0	0.7	0
Y	0	0	0	0.7	0	0	0	0	0.7	0	0	0	0	0	0.7	1.4	0.7	0	1.4	0

Sequence 3 – Formula 3

	A	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**K**	**L**	**M**	**N**	**P**	**Q**	**R**	**S**	**T**	**V**	**W**	**Y**
A	0.8	0	0	0	1.1	2.6	0	1.1	0	0	0	1.8	2.3	0	0	0	0	0	0	0
C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	1.1	0	0	2.1	0	0.6	0	0	2.1	0	0	0	0	2.5	2.5	0.9	0	0	2.1	0
E	1.5	0	2.1	0	2.1	1.6	0	0	0	0	0	0	0	0	0	1.2	0	0	0	0
F	3.4	0	0	2.1	0	0	0	1.6	0	0	0	0	0	0	0	0.9	0	2.1	0	1.6
G	0.	0	1.	0	0	0.	0	1.	0.	1.	4.	0	0	1	1	0.	0.8	1.	0	1.

	9		8			2		2	8	6	8					3		6		2
H	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3.6	0	0	0	6.2
I	0	0	0	4.2	0	1.2	6.2	0	2.1	0	0	0	3.1	0	0	0	0	0	0	0
K	0	0	0	0	2.1	0	8.3	2.1	0	0	0	0	4.2	3.3	0	1.2	0	0	0	0
L	0	0	0	0	1.4	1.1	0	0	3.7	0	0	2.2	0	0	0	1.6	1.9	0	0	0
M	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16.7	0	0	0
N	1.8	0	2.5	0	2.5	1	0	0	0	2.2	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	3.1	0	0	3.1	0	0	0	5	0	0	0	0	0	4.2	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	4	5	0	0	2.9	3.3	0	0	0
R	1.8	0	0	0	0	0	0	2.5	0	4.4	0	0	0	0	0	0	0	3.3	0	0
S	0	0	0	0	0.9	0.7	0	0.9	0	2.4	0	1.4	0	0	0	0.5	1.2	1.2	2.4	0.9
T	0	0	0	0	0	1.6	0	0	0	0	0	0	0	0	3.3	1.2	0	0	0	4.2
V	1.5	0	0	0	2.1	1.6	0	0	0	0	0	0	0	0	3.3	0	0	0	0	2.1
W	0	0	6.2	2.8	0	0	0	0	0	0	0	0	0	3.3	0	0	0	0	2.8	0
Y	0	0	0	2.1	0	0	0	0	2.1	0	0	0	0	0	2.5	1.8	2.1	0	4.2	0

To get the 10 top preferred pairs, I have written another code.

Algorithm –

- The 20*20 table, which is a list of lists, is converted to one big list.
- 10 largest values from this list is calculated.
- At each point, where the program finds a max_value, its index from the 20*20 table is found out (c,d). The corresponding Amino acid pair is appended to final_list[] and is printed

CODE-

```
def Nmaxelements(seq):
    AA_all
=['A','C','D','E','F','G','H','I','K','L','M','N','P','Q','R','S','T','V','W','Y'
]
    list1,list2,list3 = pair_pref(seq)
    listmega = [list1,list2,list3]
    print('For the given sequence, top preferred pairs for the given formulae are
listed below -')
    for listi in listmega:
        list1 = []
        for i in range(20):
```

```
            for j in range(20):
                list1.append(listi[i][j])

        final_list = []

        for i in range(10):
            max1 = list1[0]
            for j in range(len(list1)):
                if list1[j] > max1:
                    max1 = list1[j]
                    m = j
            c=0
            d = list1.index(max1)
            while d>19:
                c+=1
                d=d-20

            pair = AA_all[c]+AA_all[d]
            final_list.append(pair)
            list1[m]=0
        print('Top preferred pairs are - ')
        print(final_list)
        print()

Nmaxelements(seq1)
Nmaxelements(seq2)
Nmaxelements(seq3)
```

OUTPUT –

Sequence 1:

```
For the given sequence, top preferred pairs for the given formulae are listed below -
Top preferred pairs are -
['CF', 'FN', 'DE', 'EY', 'GC', 'GI', 'NR', 'VG', 'YD', 'WT']

Top preferred pairs are -
['AA', 'AA', 'AA', 'AA', 'AA', 'AA', 'AA', 'AA', 'AA', 'AA']

Top preferred pairs are -
['CF', 'FN', 'DE', 'GC', 'GI', 'NR', 'VG', 'IK', 'KQ', 'PV']
```

Sequence 2:

```
For the given sequence, top preferred pairs for the given formulae are listed below -
Top preferred pairs are -
['RT', 'TQ', 'VM', 'RQ', 'GL', 'KF', 'PD', 'QP', 'AA', 'AA']

Top preferred pairs are -
['AA', 'AA', 'AA', 'AA', 'AA', 'AA', 'AA', 'AA', 'AA', 'AA']

Top preferred pairs are -
['RT', 'TQ', 'KF', 'RQ', 'PD', 'QP', 'YV', 'QF', 'LE', 'LY']
```

Sequence 3:

```
For the given sequence, top preferred pairs for the given formulae are listed below -
Top preferred pairs are -
['WD', 'AG', 'FA', 'IE', 'MT', 'RL', 'TY', 'YW', 'LK', 'SL']

Top preferred pairs are -
['AG', 'FA', 'GD', 'GL', 'SL', 'WD', 'EG', 'GA', 'GI', 'GV']

Top preferred pairs are -
['MT', 'KH', 'HY', 'IH', 'WD', 'PN', 'QP', 'GM', 'RL', 'IE']
```

Question 5 –

Algorithm

- Average hydrophobicity is calculated by summing the "Hgm" values for each amino acid from the database, and dividing by the total number of amino acids present.
- Helical contact area is calculated by summing the "Ca" values for each amino acid from the database.
- Total non-bonded energy is calculated by summing the "Et" values for each amino acid from the database.

CODE –

```
def properties(seq):
    Hgm = {'A': 13.85, 'D': 11.61, 'C': 15.37, 'E': 11.38, 'F': 13.93,
           'G': 13.34, 'H': 13.82, 'I': 15.28, 'K': 11.58, 'L': 14.13,
           'M': 13.86, 'N': 13.02, 'P': 12.35, 'Q': 12.61, 'R': 13.10,
           'S': 13.39, 'T': 12.70, 'V': 14.56, 'W': 15.48, 'Y': 13.88}

    Ca = {'A': 20.00, 'D': 26.00, 'C': 25.00, 'E': 33.00, 'F': 46.00,
          'G': 13.00, 'H': 37.00, 'I': 39.00, 'K': 46.00, 'L': 35.00,
          'M': 43.00, 'N': 28.00, 'P': 22.00, 'Q': 36.00, 'R': 55.00,
          'S': 20.00, 'T': 28.00, 'V': 33.00, 'W': 61.00, 'Y': 46.00}

    Et = {'A': 1.90, 'D': 1.52, 'C': 2.04, 'E': 1.54, 'F': 1.86,
          'G': 1.90, 'H': 1.76, 'I': 1.95, 'K': 1.37, 'L': 1.97,
          'M': 1.96, 'N': 1.56, 'P': 1.70, 'Q': 1.52, 'R': 1.48,
          'S': 1.75, 'T': 1.77, 'V': 1.98, 'W': 1.87, 'Y': 1.69}

    n = len(seq)
    avg_Hgm = 0
    ca = 0
    et = 0

    for i in range(n):
        avg_Hgm+=Hgm[seq[i]]
        ca+=Ca[seq[i]]
        et+=Et[seq[i]]

    avg_Hgm = avg_Hgm/n

    print('Average hydrophobicity of the sequence = %f.' %avg_Hgm)
    print('Helical contact area of the sequence = %f.' %ca)
    print('Total non-bonded energy of the sequence = %f.\n' %et)

seq1 = 'RATPTRWPVGCFNRPWTKWSYDEALDGIKAAGYAWTGLLTASKPSLHHATATPEYLAALKQKSRHAA'
seq2 = 'AAAVMMGLAAIGAAIGIGILGGKFLEGAARQPDLIPLLRTQFFIVMGLVDAIPMIAVGLGLYVMFAVA'
```

```
seq3 =
'QSGMTYRLGLSWDWDKSWWQTSTGRLTGYWDAGYTYWEGGDEGAGKHSLSFAPVFVYEFAGDSIKPFIEAGIGVAAFSGT
RVGDQNLGSSLNFEDRIGAGLKFANGQSVGVRAIHYSNAGLKQPNDGIESYSLFYKIPI'

properties(seq1)
properties(seq2)
properties(seq3)

OUTPUT-
```

```
Average hydrophobicity of the sequence = 13.352537.
Helical contact area of the sequence = 2156.000000.
Total non-bonded energy of the sequence = 117.740000.

Average hydrophobicity of the sequence = 13.771618.
Helical contact area of the sequence = 2067.000000.
Total non-bonded energy of the sequence = 126.660000.

Average hydrophobicity of the sequence = 13.406259.
Helical contact area of the sequence = 4350.000000.
Total non-bonded energy of the sequence = 245.450000.
```

Analyses –

- All the given sequences have comparable hydrophobicity. This is due to the high composition of Alanine/other aliphatic amino acids in all the polypeptide sequences.
- The helical contact area is very high for sequence 3 compared to sequence 1 and 2. This is because, sequence 3 might exist in a right-handed alpha-helical structure, while other peptides could take up a beta-sheet conformation.
- Total non-bonded energy is also high for sequence 3 compared to sequence 1 and 2. High values of "Et" for Gly and Ser results in high value of total non-bonded energy for sequence 3.