Question 1

Link to the table. Files are also attached with the pdf.

Question 2

			Set1			
	Lowest				Highest	
Position	**Residue**	**Score**		**Position**	**Residue**	**Score**
117	K	-1.846220219		1	:	0
113	T	-1.720193459		3	L	0
70	L	-1.672625446		4	S	0
73	S	-1.594166699		7	D	0
84	E	-1.414279065		8	K	0
133	G	-1.414279065		17	K	0
36	V	-1.366711052		26	G	0
35	V	-1.294545166		30	L	0
9	A	-1.168518406		32	R	0
6	K	-1.159588814		38	P	0
			Set2			
	Lowest				Highest	
Position	**Residue**	**Score**		**Position**	**Residue**	**Score**
33	T	-1.889159164		1	M	0
64	Q	-1.831020481		2	A	0
147	A	-1.831020481		12	N	0
27	E	-1.735126457		14	K	0
23	V	-1.676987774		16	N	0
36	H	-1.676987774		43	V	0
49	L	-1.676987774		68	Q	0
93	S	-1.676987774		69	N	0
145	R	-1.676987774		76	G	0
196	L	-1.676987774		77	A	0

Question 3

In [29]:

```
from DnaUtils import readClustal
ids, seqs = readClustal("set2_MSA.mafft")
alignments = [x for x in zip(*seqs)]
header = seqs[0].replace('-',':')
mat = []

for row in open("Blossum62.txt").read().split('\n'):
    mat.append([char for char in row.split(" ") if char != ''])
mat = mat[1:]
order = list(zip(*mat))[0]
mat = list(zip(*mat))[1:]
order_mat = [a+b for a in order for b in order]
temp = []
for x in mat:
```

```
    temp += x
temp = list(map(int,temp))
blossum62= dict(zip(order_mat,temp))

def freq(seq):
    from collections import Counter
    residues = Counter(seq)
    if '-' in residues:
        residues.pop('-')
    n = sum(residues.values())
    residues = {x :y/n for x,y in residues.items()}
    return Counter(residues)

def overall_freq(seqs):
    return freq(''.join(seqs))

overall_f = overall_freq(seqs)

def entropy(seq):
    from math import log
    residues = freq(seq)
    return sum([y*log(y) for x,y in residues.items()])

def variance(seq):
    C = freq(seq)
    return (sum([(C[residue] - overall_f[residue])**2 \
                 for residue in overall_f]))**0.5

def sum_of_pairs(seq):
    x = freq(seq)
    return sum([x[seq1] * x[seq2] * blossum62[seq1+seq2] for seq1 in x for

def calculate(seqs, method = 'entropy'):
    if method == 'sum_of_pairs':
        func = sum_of_pairs
    elif method == 'variance':
        func = variance
    elif method == 'entropy':
        func = entropy
    temp = []
    for i,(seq,h) in enumerate(zip(alignments,header), start = 1):
        temp.append(f'{i},{h},{func(seq)}')
    return '\n'.join(["No.,Residue,Score"] + temp)
```

Question 4

Link to comparision table

- For set 1, all scores are the same
- For set 2, 13 positions are different: 72, 162, 163, 164, 165, 258, 259, 260, 261, 262, 263, 264,

Question 5

SET1

Clustal			MAFFT			MUSCLE		
No.	Residue	Score	No.	Residue	Score	No.	Residue	Score
9	A	-1.168518406	9	T	-1.168518406	9	A	-1.168518406
11	V	-0.5859526183	11	V	-0.5859526183	11	V	-0.5859526183
20	K	-1.033562054	20	A	-1.033562054	20	K	-1.033562054
22	A	-0.4741393131	22	A	-0.4741393131	22	A	-0.4741393131
30	L	0	30	L	0	30	L	0

SET2

Clustal			MAFFT			MUSCLE		
No.	Residue	Score	No.	Residue	Score	No.	Residue	Score
9	A	-0.3767701613	9	V	-0.3767701613	9	A	-0.3767701613
11	A	-0.3767701613	11	G	-0.3767701613	11	A	-0.3767701613
20	S	-1.522955068	20	Q	-1.522955068	20	S	-1.522955068
22	L	-1.060856947	22	L	-1.060856947	22	L	-1.060856947
30	N	-0.3488320958	30	N	-0.3488320958	30	N	-0.3488320958

Question 6

Link to file containting conservation scores. File is also attached with this pdf.

Alignment details

The average number of replacements between any two sequences in the alignment; A distance of 0.01 means that on average, the expected replacement for every 100 positions is 1.

1. Average pairwise distance : 0.99628
2. Lower bound : 0.10971
3. Upper bound : 1.89437