Q1: Initialise a variable with a signal peptide sequence, change the case and print it. Also print the length of the sequence. (Hint- use string manipulation functions)

```
signalpeptide = "mkialrkllllalfl"
signalpeptide = signalpeptide.upper()
print ("There are %d amino acids present in the given signal peptide :%s " %(len(signalpe
```

```
There are 15 amino acids present in the given signal peptide :MKIALRKLLLLALFL
```

Q2: Check the existence of a particular amino acid in a peptide sequence

```
peptide = "MKKIEKLTEMANIIGG"

#K - lysine
if('K' in peptide):
  print("The amino acid lysine is present")
else:
   print("The amino acid lysine is absent")

#R - arginine
if('R' in peptide):
  print("The amino acid arginine is present")
else:
   print("The amino acid arginine is absent")
```

```
The amino acid lysine is present
The amino acid arginine is absent
```

Q3: Count the number of Nucleotides in a DNA Sequence (Human Insulin)

> - Using a for loop
> - Using a while loop
> - Using python library function

```
#dna sequence of insulin gene
import re;
dna = """AGCCCTCCAGGACAGGCTGCATCAGAAGAGGCCATCAAGCAGGTCTGTTCCAAGGGCCTTTGCGTCAGGT
GGGCTCAGGATTCCAGGGTGGCTGGACCCCAGGCCCCAGCTCTGCAGCAGGGAGGACGTGGCTGGGCTCG
TGAAGCATGTGGGGGTGAGCCCAGGGGCCCCAAGGCAGGGCACCTGGCCTTCAGCCTGCCTCAGCCCTGC
CTGTCTCCCAGATCACTGTCCTTCTGCCATGGCCCTGTGGATGCGCCTCCTGCCCCTGCTGGCGCTGCTG
GCCCTCTGGGGACCTGACCCAGCCGCAGCCTTTGTGAACCAACACCTGTGCGGCTCACACCTGGTGGAAG
CTCTCTACCTAGTGTGCGGGGAACGAGGCTTCTTCTACACACCCAAGACCCGCCGGGAGGCAGAGGACCT
GCAGGGTGAGCCAACTGCCCATTGCTGCCCCTGGCCGCCCCCAGCCACCCCCTGCTCCTGGCGCTCCCAC
CCAGCATGGGCAGAAGGGGGCAGGAGGCTGCCACCCAGCAGGGGGTCAGGTGCACTTTTTTAAAAAGAAG
TTCTCTTGGTCACGTCCTAAAAGTGACCAGCTCCCTGTGGCCCAGTCAGAATCTCAGCCTGAGGACGGTG
TTGGCTTCGGCAGCCCCGAGATACATCAGAGGGTGGGCACGCTCCTCCCTCCACTCGCCCCTCAAACAAA
TGCCCCGCAGCCCATTTCTCCACCCTCATTTGATGACCGCAGATTCAAGTGTTTTGTTAAGTAAAGTCCT
GGGTGACCTGGGGTCACAGGGTGCCCCACGCTGCCTGCCTCTGGGCGAACACCCCATCACGCCCGGAGGA
GGGCGTGGCTGCCTGCCTGAGTGGGCCAGACCCCTGTCGCCAGGCCTCACGGCAGCTCCATAGTCAGGAG
ATGGGGAAGATGCTGGGGACAGGCCCTGGGGAGAAGTACTGGGATCACCTGTTCAGGCTCCCACTGTGAC
GCTGCCCCGGGGCGGGGGAAGGAGGTGGGACATGTGGGCGTTGGGGCCTGTAGGTCCACACCCAGTGTGG
GTGACCCTCCCTCTAACCTGGGTCCAGCCCGGCTGGAGATGGGTGGGAGTGCGACCTAGGGCTGGCGGGC
AGGCGGGCACTGTGTCTCCCTGACTGTGTCCTCCTGTGTCCCTCTGCCTCGCCGCTGTTCCGGAACCTGC
TCTGCGCGGCACGTCCTGGCAGTGGGGCAGGTGGAGCTGGGCGGGGGCCCTGGTGCAGGCAGCCTGCAGC
CCTTGGCCCTGGAGGGGTCCCTGCAGAAGCGTGGCATTGTGGAACAATGCTGTACCAGCATCTGCTCCCT
CTACCAGCTGGAGAACTACTGCAACTAGACGCAGCCCGCAGGCAGCCCCACACCCGCCGCCTCCTGCACC
GAGAGAGATGGAATAAAGCCCTTGAACCAGC"""
dna = re.sub('\s','',dna);
```

```
# for loop

dna = dna.upper()
A=T=G=C=0

for i in range (len(dna)):
  if dna[i] == 'A':
    A+=1
  if dna[i] == 'T':
    T+=1
  if dna[i] == 'G':
    G+=1
  if dna[i] == 'C':
    C+=1

print("Number of bases: A : ", A)
print("Number of bases: T : ", T)
print("Number of bases: G : ", G)
print("Number of bases: C : ", C)
```

```
Number of bases: A :  247
Number of bases: T :  259
Number of bases: G :  456
Number of bases: C :  469
```

```
# while loop

dna = dna.upper()
A=T=G=C=0
i=0
while i < len(dna):
    if dna[i] == 'A':
      A+=1
    if dna[i] == 'T':
      T+=1
    if dna[i] == 'G':
      G+=1
    if dna[i] == 'C':
      C+=1
    i+=1

print("Number of bases: A : ", A)
print("Number of bases: T : ", T)
print("Number of bases: G : ", G)
print("Number of bases: C : ", C)
```

```
Number of bases: A :  247
Number of bases: T :  259
Number of bases: G :  456
Number of bases: C :  469
```

```
# library function

dna = dna.upper()
total = len(dna)
A = dna.count('A')
T = dna.count('T')
G = dna.count('G')
C = dna.count('C')

print(f"total {total} nucleotides")
print("Frequency of nucleotides are :\n")
print("Nucleotide | Count\n------------------")
print(f"    A      | {A}\n    T      | {T}\n    G      | {G}\n    C      | {C}")
```

```
total 1431 nucleotides
Frequency of nucleotides are :

Nucleotide | Count
------------------
    A      | 247
    T      | 259
    G      | 456
    C      | 469
```

Q4: Calculate the 'GC Content' of a DNA Sequence

```
G = dna.count('G')
C = dna.count('C')
total = len(dna)

print(f"total nucleotides present : {total}")
gc_content = ((G+C)/total)*100
print(f"GC content in percentage : {gc_content}")
```

```
total nucleotides present : 1431
GC content in percentage : 64.64011180992313
```

Q5: Check the existence of a 'TATA Box' in a DNA sequence

```
import re
pattern = "TATA[AT]A[AT]"
dna_sequence = "TAGACGTTATAAAATGCCCTCAGATAGCCG"
match_object = re.search(pattern, dna_sequence)
if match_object:
    print("The TATA Box is present at position:", match_object.start())
else:
    print("The TATA Box is not found in the given DNA sequence.")
```

```
The TATA Box is present at position: 7
```

Q6: Problem : Find the Reverse Complement of a DNA String

```
dna = "ATGCGTACCGTCAGATCGATCGATCGATCGTAGCTAGCATCGATCGATCGATCGATCGATCGATCGATCGATCGTAGC
cdna = ""

for i in range(len(dna)):
  if dna[i] == 'A':
    cdna += 'T'
  if dna[i] == 'T':
    cdna += 'A'
  if dna[i] == 'G':
    cdna += 'C'
  if dna[i] == 'C':
    cdna += 'G'

rcdna = cdna[::-1]
print("Double stranded DNA is :")
print(dna)
print(rcdna)
```

```
Double stranded DNA is :
ATGCGTACCGTCAGATCGATCGATCGATCGTAGCTAGCATCGATCGATCGATCGATCGATCGATCGATCGATCGTAGCTAG
CGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATC
```

Q7: Convert a DNA sequence into an mRNA Sequence

```
dna = """ATGCGTACCGTCAGATCGATCGATCGATCGTAGCTAGCATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGTA
"""

mrna=""
for base in dna:
  if(base == 'A'):
    mrna += 'U'
  if(base == 'T'):
    mrna += 'A'
  if(base == 'G'):
    mrna += 'C'
  if(base == 'C'):
    mrna += 'G'



print ("The mRNA sequence is: \n",mrna);
```

```
The mRNA sequence is:
 UACGCAUGGCAGUCUAGCUAGCUAGCUAGCAUCGAUCGUAGCUAGCUAGCUAGCUAGCUAGCUAGCUAGCUAGCAUCGAL
```

Q8: Check the existence of a start codon in the mRNA sequence

```
import re

mrna = "UUCUACAAUGCCUACCUAACA"
if re.search("AUG", mrna):
    print("The mRNA sequence contains the start codon AUG.")
    position = re.search("AUG", mrna)
    print("AUG occurs at position:", position.start())
else:
    print("The start codon AUG is not found in the mRNA sequence.")
```

```
The mRNA sequence contains the start codon AUG.
AUG occurs at position: 7
```

Q9: Check the existence of stop codon in the mRNA sequence

```
import re
if(re.search("UAA|UAG|UGA",mrna)):
  print("A stop codon is found in the mRNA sequence ")
  pos = re.search("UAA|UAG|UGA",mrna)
  print("The occurrence of ",pos.group()," is at : ",pos.start())
else:
  print("The stop codon not found in the mRNA sequence")
```

```
A stop codon is found in the mRNA sequence
The occurrence of  UAG  is at :  14
```

Q10: Find positions of the start and stop codon present in an mRNA sequence. Extract the coding region using the start and stop codons

```
import re
mrna = "AUGCCUACCUAA"
exp = "AUG\w+UAA|UAG|UGA"
x= re.search(exp,mrna)
str = x.group()[:-3]
print("The coding region is (with stop codon) : ",x.group())
print("The coding region is : ",str)
```

```
The coding region is (with stop codon) :  AUGCCUACCUAA
The coding region is :  AUGCCUACC
```

Q11: Use a python dictionary to convert an peptide chain from its one letter codon representation to three letter codon representation

```
threeLetterCode = {'A':'ALA','C':'CYS','D':'ASP','E':'GLU','F':'PHE','G':'GLY',
                     'H':'HIS','I':'ILE','K':'LYS','L':'LEU','M':'MET','N':'ASN',
                     'P':'PRO','Q':'QLN','R':'ARG','S':'SER','T':'THR','V':'VAL',
                     'W':'TRP','Y':'TYR'}
peptide1 = "MASKATLLLAFTLLFATCIA"
peptide2 =""
for i in peptide1:
     peptide2 +=threeLetterCode[i]
print("Polypeptide chain")
print("1 letter code  : ",peptide1)
print("3 letter code  : ",peptide2)
```

```
Polypeptide chain
1 letter code  :  MASKATLLLAFTLLFATCIA
3 letter code  :  METALASERLYSALATHRLEULEULEUALAPHETHRLEULEUPHEALATHRCYSILEALA
```

Q12 : calculates the molecular weight of a protein based on weight of individual amino acids (Given as a dictionary)

```
peptide = "VPQLRHYGLIASTKTRVLFQ"
proteinWt = {'A':89, 'V':117, 'L':131, 'I':131, 'P':115,
 'F':165, 'W':204, 'M':149, 'G':75, 'S':105,
 'C':121, 'T':119, 'Y':181, 'N':132, 'Q':146,
'D':133, 'E':147, 'K':146, 'R':174, 'H':155}
total = 0

for aa in peptide:
  total += proteinWt.get(aa.upper(), 0)
total -= (18 * (len(peptide) - 1))
print(f"The net weight of the Protein : {total}")
```

```
The net weight of the Protein : 2325
```