

Practical No: 1

Aim: Write a Python/Java code to perform pairwise alignment. Take 2 sequences from user and calculate the score.

Code:

```
se1=input("Enter the first sequence::")
se2=input("Enter the second sequence::")
seq1=list(se1)
seq2=list(se2)
score=[]
```

```
def Pairwise_alignment(a,b):
```

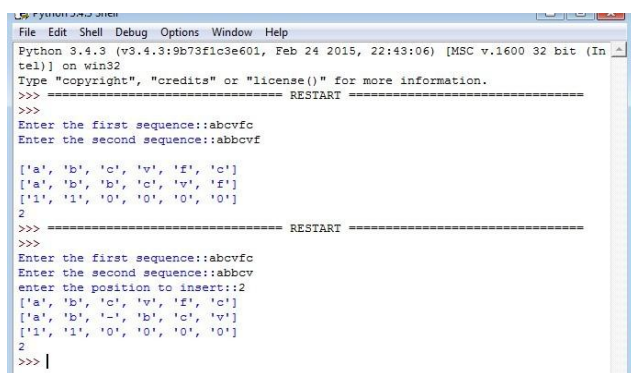
```
    gap(a,b)
    print(a)
    print(b)
    value=0
    length=len(a)
    for i in range(0,length):
        if(a[i]==b[i]):
            score.append('1')
            value=value+1
        else:
            score.append('0')
    print(score)
    print(value)
```

```
def gap(a,b):
```

```
    if(len(a)==len(b)):
        print()
    else:
        k=int(input("enter the position to insert::"))
        if (len(a)<len(b)):
            a.insert(k,'-')
        else:
            b.insert(k,'-')
    return(a,b)
```

```
Pairwise_alignment(seq1,seq2)
```

Output:



```

Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
Enter the first sequence::abcvfc
Enter the second sequence::abcvf

['a', 'b', 'c', 'v', 'f', 'c']
['a', 'b', 'b', 'c', 'v', 'f']
['1', '1', '0', '0', '0', '0']
2
>>> ===== RESTART =====
>>>
Enter the first sequence::abcvfc
Enter the second sequence::abcv
enter the position to insert::2
['a', 'b', 'c', 'v', 'f', 'c']
['a', 'b', '-', 'b', 'c', 'v']
['1', '1', '0', '0', '0', '0']
2
>>> |

```

Practical No: 2

Aim: Write a Python/Java code to find the identity value of a given sequences. Take the sequence from user.

Code:

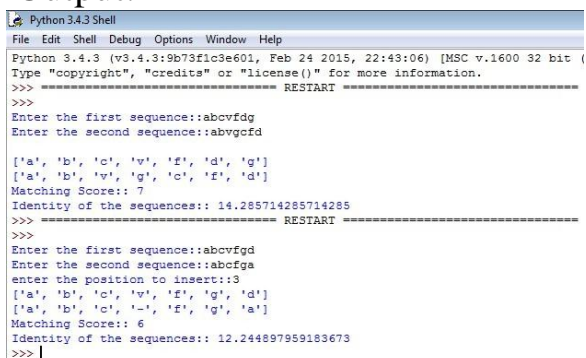
```

se1=input("Enter the first sequence::")
se2=input("Enter the second sequence::")

seq1=list(se1)
seq2=list(se2)
def find_identity(a,b):
    gap(a,b)
    print(a)
    print(b)
    score=0
    length=len(a)
    total_elements=len(a)*len(b)
    for i in range(0,length):
        for j in range(0,length):
            if(a[i]==b[j]):
                score=score+1
    identity=(score/total_elements)*100
    print("Matching Score::",score)
    print("Identity of the sequences::",identity)
def gap(a,b):
    if(len(a)==len(b)):
        print()
    else:
        k=int(input("enter the position to insert gap ::"))
        if (len(a)<len(b)):
            a.insert(k,'-')
        else:
            b.insert(k,'-')

```

```
    return(a,b)  
find_identity(seq1,seq2)
```

Output:


```

Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
Enter the first sequence::abcvfdg
Enter the second sequence::abvgcfd

['a', 'b', 'c', 'v', 'f', 'd', 'g']
['a', 'b', 'v', 'g', 'c', 'f', 'd']
Matching Score:: 7
Identity of the sequences:: 14.285714285714285
>>> ===== RESTART =====
>>>
Enter the first sequence::abcvfdg
Enter the second sequence::abcfga
enter the position to insert::3
['a', 'b', 'c', 'v', 'f', 'g', 'd']
['a', 'b', 'c', '-', 'f', 'g', 'a']
Matching Score:: 6
Identity of the sequences:: 12.244897959183673
>>> |

```

Practical No: 3

Aim: Write a Python/Java code to find the Similarity value of a given sequences. Take the sequence from user.

Code:

```

sequence_one=input("Enter the first sequence: ")
sequence_two=input("Enter the second sequence: ")
how_many=int(input("How many elements for similarity condition?"))
similarities=[]
for i in range(0,how_many):
    a=input("Enter an element: ")
    c=int(input("How many elements is it similar to? "))
    similarities.append([])
    similarities[i].append(a)

    for j in range(0,c):
        b=input("What is it similar to? ")

        similarities[i].append(b)

def compare(o,t,s):
    print(o)
    print(t)
    print(s)
    #checking if similar
    score=0
    for i in range(len(o)):
        for j in range(len(s)):

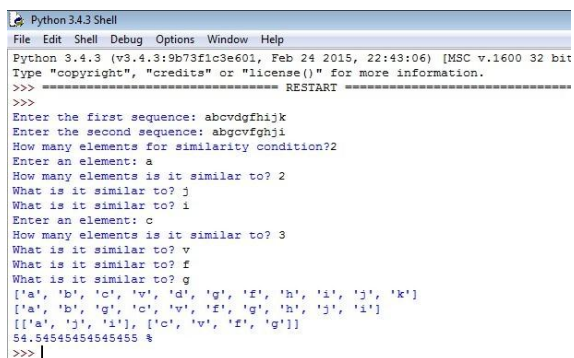
            if o[i] in s[j] and t[i] in s[j] and o[i] != t[i]:
                score+=1
    #calculating similarity

```

```
similarity= (score*100)/len(o)  
return similarity
```

```
print(compare(list(sequence_one),list(sequence_two),similarities,"%")
```

Output:



```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
Enter the first sequence: abcvdgfhijk
Enter the second sequence: abgcvfghji
How many elements for similarity condition?2
Enter an element: a
How many elements is it similar to? 2
What is it similar to? j
What is it similar to? i
Enter an element: c
How many elements is it similar to? 3
What is it similar to? v
What is it similar to? f
What is it similar to? g
['a', 'b', 'c', 'v', 'd', 'g', 'f', 'h', 'i', 'j', 'k']
['a', 'b', 'g', 'c', 'v', 'f', 'g', 'h', 'j', 'i']
[['a', 'j', 'i'], ['c', 'v', 'f', 'g']]
54.54545454545455 %
>>> |
```

Practical No: 4

Aim: Enter genome of five different organism and write a python/java program to find consensus sequence using Multiple Sequence Alignment (MSA) technique.

Code:

```
import java.io.*;

import java.util.*;

public class Consensus

{

    public static void main(String str[]) throws IOException

    {

        int n, i,j,k,count;

        String seq[],cons[];

        ArrayList<Integer> a = new ArrayList<Integer>();

        ArrayList s = new ArrayList();

        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

        System.out.println("Enter the no of Sequences");

        n=Integer.parseInt(br.readLine());

        seq=new String[n];
```

```
System.out.println("Enter sequences");  
for(i=0;i<n;i++)
```

```
seq[i]=br.readLine();
cons=new String[seq[0].length()];
for(j=0;j<seq[0].length();j++)
cons[j]=" ";
for(j=0;j<seq[0].length();j++)
{
    a.clear();
    s.clear();
    for(i=0;i<n;i++)
    {
        count=1;
        for(k=i+1;k<n;k++)
        {

            if(seq[i].charAt(j)==seq[k].charAt(j))
                count++;

        }
        System.out.println("count="+count);
        a.add(count);
        s.add(seq[i].charAt(j));
    }
}

/**Updated Snippet 1**/
Set<String> set = new HashSet<>(s);
ArrayList setlist = new ArrayList(set);
Collections.sort(setlist);

if (setlist.contains('-') && setlist.size()==2){
    cons[j]+="-"+setlist.get(1);
}

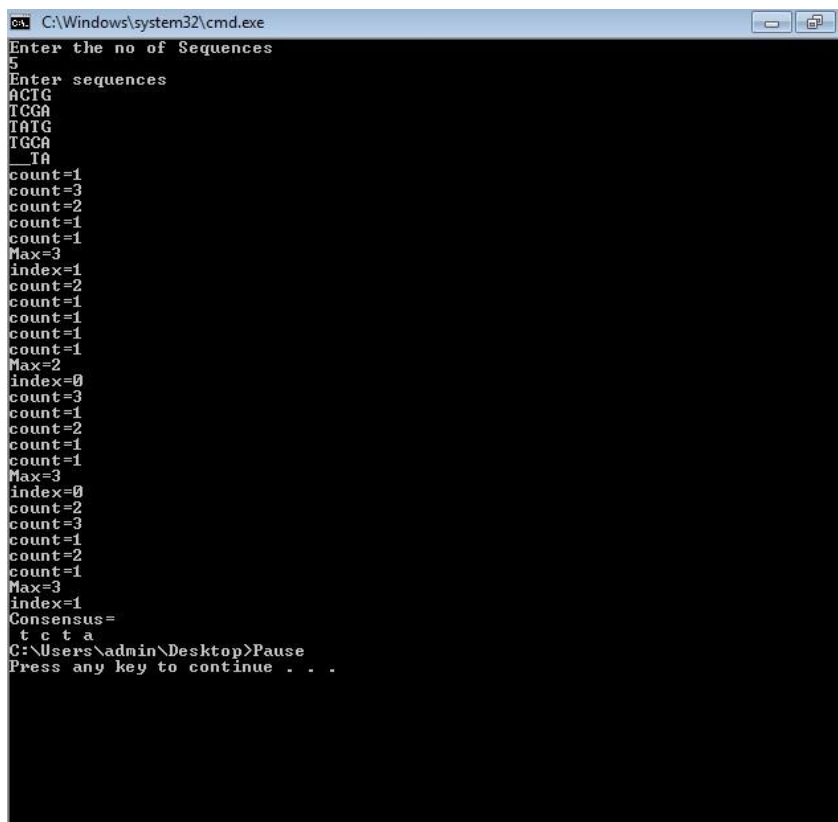
else if (setlist.size()==1){
    cons[j]+="-"+setlist.get(0);
```



```
}  
else{  
    int m = Collections.max(a);
```

```
int index=a.indexOf(m);
System.out.println("Max="+m);
cons[j]+=s.get(index);
System.out.println("index="+index);
for(i=index+1;i<a.size();i++)
{
    if(a.get(i)==m)
        cons[j]+="/" +s.get(i);
}
}
}
System.out.println("Consensus=");
for(j=0;j<seq[0].length();j++){
    /**Updated Snippet 2**/
    if(cons[j].length()==2)
        System.out.print(cons[j].toLowerCase());
    else if(cons[j].length()==3)
        System.out.print(cons[j].replace("-", ""));
    else
        System.out.print(cons[j]);
}
}
}
```

Output:



```
C:\Windows\system32\cmd.exe
Enter the no of Sequences
5
Enter sequences
ACTG
TCGA
TAAG
TCGA
_T A
count=1
count=3
count=2
count=1
count=1
Max=3
index=1
count=2
count=1
count=1
count=1
count=1
Max=2
index=0
count=3
count=1
count=2
count=1
count=1
Max=3
index=0
count=2
count=3
count=1
count=2
count=1
Max=3
index=1
Consensus=
t c t a
C:\Users\admin\Desktop>Pause
Press any key to continue . . .
```

Practical No: 5

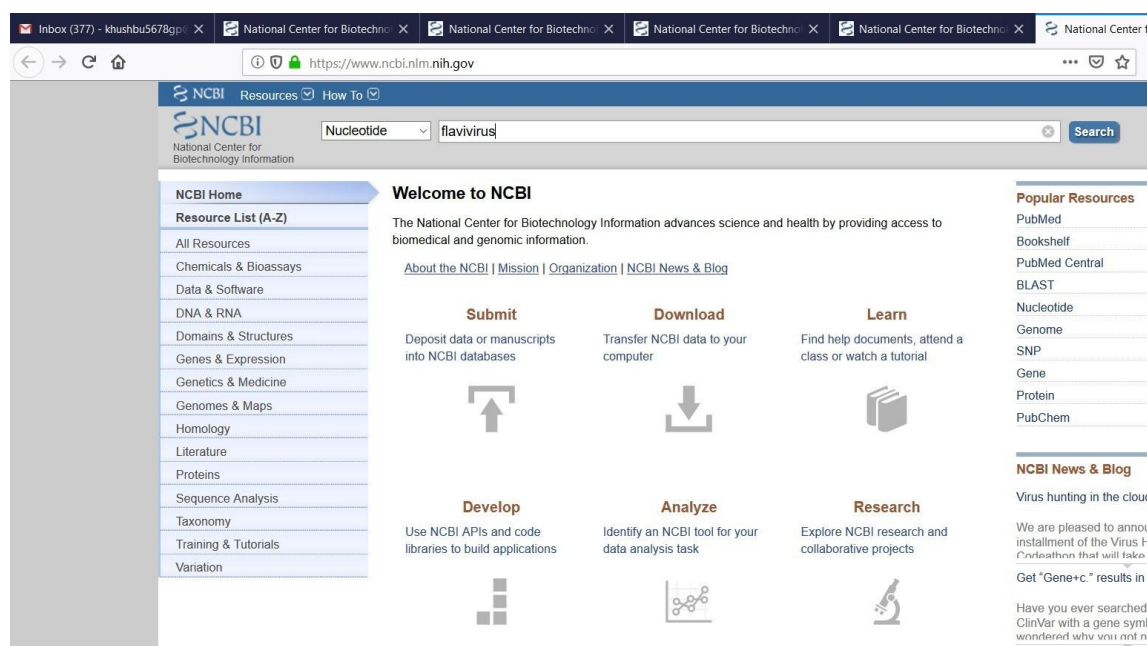
Aim: Perform a BLAST search on a specific gene sequence of a specify organism.

Steps:

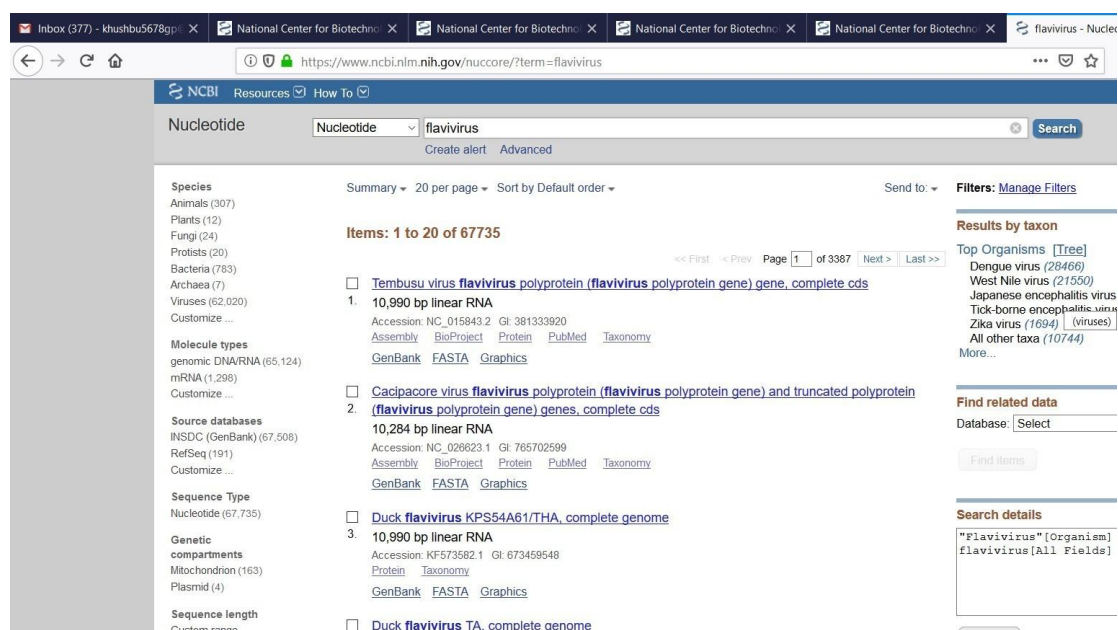
Go to the National Center for Biotechnology Information Site

<https://www.ncbi.nlm.nih.gov/>

Select Nucleotide from All Databases and find any organism in a search bar.



The screenshot shows the NCBI homepage with a search bar at the top containing 'flavivirus'. The left sidebar lists various resources like 'Nucleotide', 'Protein', and 'Gene'. The main content area features a 'Welcome to NCBI' message and several interactive buttons: 'Submit', 'Download', 'Learn', 'Develop', 'Analyze', and 'Research'. A 'Popular Resources' section on the right lists links to PubMed, Bookshelf, and other databases.



The screenshot displays the NCBI Nucleotide search results for the query 'flavivirus'. The results are sorted by Default order and show 3387 items. The first three results are listed:

- 1. 10,990 bp linear RNA**
Accession: NC_015843.2 | GI: 381333920
Assembly | BioProject | Protein | PubMed | Taxonomy
[GenBank](#) [FASTA](#) [Graphics](#)
- 2. 10,284 bp linear RNA**
Accession: NC_026623.1 | GI: 765702599
Assembly | BioProject | Protein | PubMed | Taxonomy
[GenBank](#) [FASTA](#) [Graphics](#)
- 3. 10,990 bp linear RNA**
Accession: KF573582.1 | GI: 673459548
Protein | Taxonomy
[GenBank](#) [FASTA](#) [Graphics](#)

The right sidebar shows 'Results by taxon' with a list of top organisms including Dengue virus, West Nile virus, and Japanese encephalitis virus. There is also a 'Find related data' section and a 'Search details' section.

NCBI Resources How To

Nucleotide

Advanced

GenBank

Tembusu virus flavivirus polyprotein (flavivirus polyprotein gene) gene, complete cds

NCBI Reference Sequence: NC_015843.2

[FASTA](#) [Graphics](#)

Go to:

LOCUS NC_015843 10990 bp ss-RNA linear VRL 13-AUG-2018

DEFINITION Tembusu virus flavivirus polyprotein (flavivirus polyprotein gene) gene, complete cds.

ACCESSION NC_015843 NC_016958 NC_018670

VERSION NC_015843.2

DBLINK BioProject: [PRJNA485481](#)

KEYWORDS RefSeq.

SOURCE Tembusu virus (TMUV)

ORGANISM Tembusu virus

Viruses; Riboviria; Flaviviridae; Flavivirus.

REFERENCE 1 (bases 1 to 10990)

AUTHORS Han,K., Huang,X., Li,Y., Zhao,D., Liu,Y., Zhou,X., You,Y. and Xie,X.

TITLE Complete genome sequence of goose tembusu virus, isolated from jiangnan white geese in jiangsu, china

JOURNAL Genome Announc 1 (2), E0023612 (2013)

PUBMED 23516233

REMARK Publication Status: Online-Only

2 (bases 1 to 10990)

Change region shown

Customize view

Analyze this sequence

Run BLAST

Pick Primers

Highlight Sequence Features

Find in this Sequence

Related information

Assembly

BioProject

Protein

PubMed

Taxonomy

Full text in PMC

Functional Class

Run BLAST option we have to select

Align two or more sequences

Choose Search Set

Database ☐ Human genomic + transcript ☐ Mouse genomic + transcript ☒ Others (nr etc.):

Nucleotide collection (nr/nt)

Organism Optional

Enter organism name or id--completions will be suggested

Enter organism common name, binomial, or tax id. Only 20 top taxa will be shown

Exclude Optional

☐ Models (XM/XP) ☐ Uncultured/environmental sample sequences

Limit to Optional

Sequences from type material

Entrez Query Optional

Enter an Entrez query to limit search

Program Selection

Optimize for

☒ Highly similar sequences (megablast)

☐ More dissimilar sequences (discontiguous megablast)

☐ Somewhat similar sequences (blastn)

Choose a BLAST algorithm

Search database Nucleotide collection (nr/nt) using Megablast (Optimize for highly similar sequences)

☐ Show results in a new window

[Algorithm parameters](#)

BLAST is a registered trademark of the National Library of Medicine

BLAST

The screenshot shows the NCBI Blast results page for a query sequence. The table lists 99 sequences selected, sorted by maximum score. The columns are: Description, Max Score, Total Score, Query Cover, and E value. The sequences are all complete genomes of various Tembusu virus strains and isolates, with E values of 0.0.

Description	Max Score	Total Score	Query Cover	E value
Tembusu virus strain JS804, complete genome	20064	20064	99%	0.0
Tembusu virus strain JS/2010, complete genome	20048	20048	99%	0.0
Duck egg-drop syndrome virus strain byd1, complete genome	20026	20026	99%	0.0
Tembusu virus isolate Tembusu virus strain, complete genome	20020	20020	99%	0.0
Duck Tembusu virus isolate df-2, complete genome	20020	20020	99%	0.0
Duck egg-drop syndrome virus strain JXSP, complete genome	20015	20015	99%	0.0
Tembusu virus isolate YY5, complete genome	20009	20009	99%	0.0
Tembusu virus isolate SDMS, complete genome	20009	20009	99%	0.0
Tembusu virus isolate ZJ-6, complete genome	20004	20004	99%	0.0
Tembusu virus strain AH-F10 from China, complete genome	20004	20004	99%	0.0
Duck egg-drop syndrome virus strain pigeon, complete genome	19998	19998	99%	0.0
Tembusu virus genomic RNA, complete genome, strain: TMUW-YY1Du	19998	19998	99%	0.0
Duck Tembusu virus strain BZ_2010, complete genome	19998	19998	99%	0.0
Duck egg-drop syndrome virus strain duan, complete genome	19989	19989	99%	0.0
Duck Tembusu virus strain GDLH01, complete genome	19989	19989	99%	0.0

Here the result will be display

The screenshot shows the detailed alignment for the top hit, Tembusu virus strain JS804, complete genome. The sequence ID is JF895923.2, with a length of 10990 and 1 match. The alignment shows a perfect match between the query and the subject sequence.

Score	Expect	Identities	Gaps	Strand
20295 bits(10990)	0.0	10990/10990(100%)	0/10990(0%)	Plus/Plus

Range 1: 1 to 10990

Query	Subject	Score
1 AGAAGTTCGCCTGTGTGAACCTTATCCAAACAGCTTTGGAGTAGTGCCTGTGAACGTAA	1 AGAAGTTCGCCTGTGTGAACCTTATCCAAACAGCTTTGGAGTAGTGCCTGTGAACGTAA	60
61 ACACAGTTTGAACGTTTTTGGATAGAGACAACATATGTCTAACAAAAAACAGGAAGACC	61 ACACAGTTTGAACGTTTTTGGATAGAGACAACATATGTCTAACAAAAAACAGGAAGACC	120
121 CGGCTCAGGCCGGGTGTCAATATGCTAAAGCGCGGAACGTCGCGCGGAAATCCGCTAGC	121 CGGCTCAGGCCGGGTGTCAATATGCTAAAGCGCGGAACGTCGCGCGGAAATCCGCTAGC	180
181 GCGGATAAAGAGGACGATTGATGGGTCCTGAGAGGAGCAGGACCCATAAGGTTTGTGCT	181 GCGGATAAAGAGGACGATTGATGGGTCCTGAGAGGAGCAGGACCCATAAGGTTTGTGCT	240
241 GGCTCTACTGACTTTCTTCAAGTTTACAGCCCTGAGGCCAACATTGGAATGCTGAAGAG	241 GGCTCTACTGACTTTCTTCAAGTTTACAGCCCTGAGGCCAACATTGGAATGCTGAAGAG	300
301 ATGGAAGCTGGTTGGAGTTAATGAGGCGACCAACATCTGAAAAGCTTCAAGCGTGACAT	301 ATGGAAGCTGGTTGGAGTTAATGAGGCGACCAACATCTGAAAAGCTTCAAGCGTGACAT	360
361 TGGACAGATGCTCGACGACTGAATAAGCGGAAGCGAAACGTCGGGGGGGAGTTGCTC	361 TGGACAGATGCTCGACGACTGAATAAGCGGAAGCGAAACGTCGGGGGGGAGTTGCTC	420

Practical No: 6

Aim: Write a Python/Java code to find motif in a given sequence.

Code:

```
import random

l=int(input("Enter the length of motif"))

file=open("mot.txt","r")

r=file.read()

print("Sequence",r)

size=len(r)

print("Size of the sequence",size)

pos=random.randint(0,len(r)-5)

#pos=1

print("Position",pos)

motif=r[pos:pos+l]

print("Motif",motif)

i=pos+1

while(i<=size-1):

    if(motif==r[i:i+1]):

        str1=r[i:i+1]

        print("Match motif",str1)

        file1=open("motoutput.txt","a")

        file1.write(str1+" ")

    i+=1
```

Output:

```
Enter the length of motif4

Sequence AGAAGTTCGAGAAGCCGTAGT

Size of the sequence 21

Position 0

Motif AGAA
```

Practical No: 7

BIOINFORMATICS

Aim: Perform a BLAST search on any genes sequence and write a java/python code to count the no of repetition of each nucleotide in the sequence.

Code:

```
file=open("genes.txt","r")
r=file.read()
size=len(r)
score_A=0
score_C=0
score_T=0
score_G=0
for i in range(size):
    if(r[i]=='A'):
        score_A+=1
    elif (r[i]=='C'):
        score_C+=1
    elif (r[i]=='T'):
        score_T+=1
    elif (r[i]=='G'):
        score_G+=1
print("score of A is ",score_A)
print("score of C is ",score_C)
print("score of T is ",score_T)
print("score of G is ",score_G)
```

Output:

```
score of A is 6
score of C is 4
```


score of T is 7

score of G is 6

Practical No: 8

Aim: Generate a regular expression enter three protein sequence of three different organism. Write Python/Java code to find regular expression for this sequences.

Code:

```
def gen_reg_exp(seq_list, no_of_col):  
    final_list=[]  
    for colnum in range(no_of_col):  
        collist=[]  
        for colseq in seq_list:  
            collist.append(colseq[colnum])  
        if len(set(collist))==len(collist):  
            #print(final_list)  
            final_list.append('x')  
        else:  
            if len(set(collist))==1:  
                final_list.append(collist[0])  
            else:  
                final_list.append("".join(set(collist)))  
    display_output(final_list)
```

```
def display_output(final_list):  
    print(*final_list, sep='-')  
  
no_of_seq=int(input("Enter the number of sequence: "))  
print("Enter all the sequences")  
seq_list=[]  
for _ in range(no_of_seq):  
    seq_list.append(list(map(str, input("").split())))  
gen_reg_exp(seq_list, len(seq_list[0]))
```

Output:

```
Enter the number of sequence: 4  
Enter all the sequences  
A D L G A V F A L C D R Y F Q  
S D V G P R S C F C E R F Y Q  
A D L G R T Q L R C D R Y Y Q  
A D I G Q P H S L C E R Y F Q  
SA-D-IVL-G-x-x-x-x-FRL-C-ED-R-YF-YF-Q
```

Practical No: 9

Aim: Enter six protein sequence of different organism and write a program to find a fingerprint of sequence.

Code:

```
def solve_fingerprint(seq_list, no_of_col):  
    seq_dict=dict()  
  
    for colnum in range(no_of_col):  
        counta,countc,countt,countg=0,0,0,0
```

```
for colseq in seq_list:
```

```
        if colseq[colnum]=='A':
            counta+=1
        elif colseq[colnum]=='T':
            countt+=1
        elif colseq[colnum]=='C':
            countc+=1
        elif colseq[colnum]=='G':
            countg+=1
    seq_dict[colnum]=[counta,countc,countt,countg]
display_results(seq_dict)

def display_results(seq_dict):
    print("\tA \tC \tT \tG")
    for key in seq_dict:
        print("\n",*seq_dict[key],sep="\t")

no_of_seq=int(input("Enter the number of sequence: "))
print("Enter all the sequences")
seq_list=[]
for _ in range(no_of_seq):
    seq_list.append(list(map(str, input("").split()))))
solve_fingerprint(seq_list,len(seq_list[0]))
```

Output:

Enter the number of sequence: 4

Enter all the sequences

A C T G A T G

A T C A G A A

A T A A G C A

A G T T A G C

	A	C	T	G
--	---	---	---	---

4	0	0	0
---	---	---	---

0	1	2	1
---	---	---	---

1	1	2	0
---	---	---	---

2	0	1	1
---	---	---	---

2	0	0	2
---	---	---	---

1	1	1	1
---	---	---	---

2	1	0	1
---	---	---	---