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)):

1. insert(k,'-') else:
2. insert(k,'-') return(a,b)

Pairwise\_alignment(seq1,seq2)

Output:

Graphical user interface, text, application, email

Description automatically generated

Practical No: 2

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

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)):

1. insert(k,'-') else:
2. insert(k,'-') return(a,b)

find\_identity(seq1,seq2)

Output:

Graphical user interface, text, application

Description automatically generated

Practical No: 3

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

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

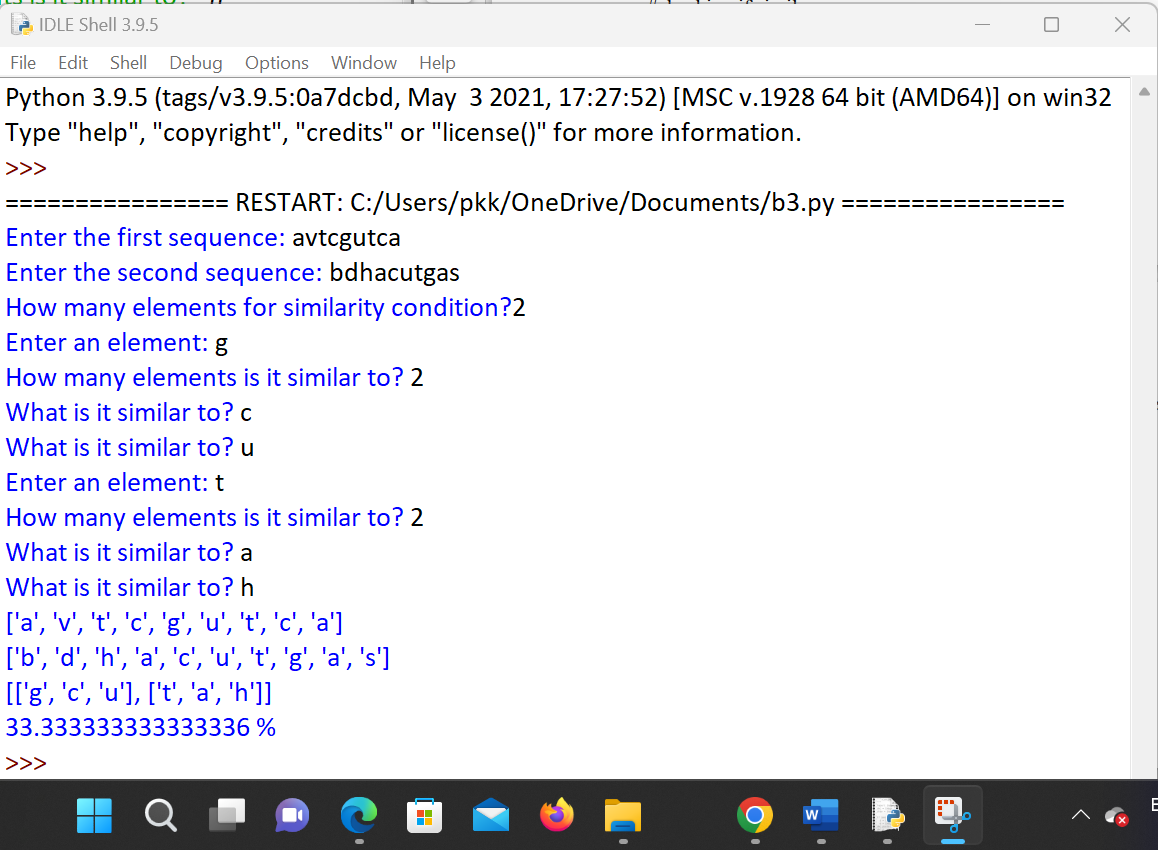
for i in range(len(o)): for j in range(len(s)):

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:



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:

Text

Description automatically generated

Practical No: 5

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

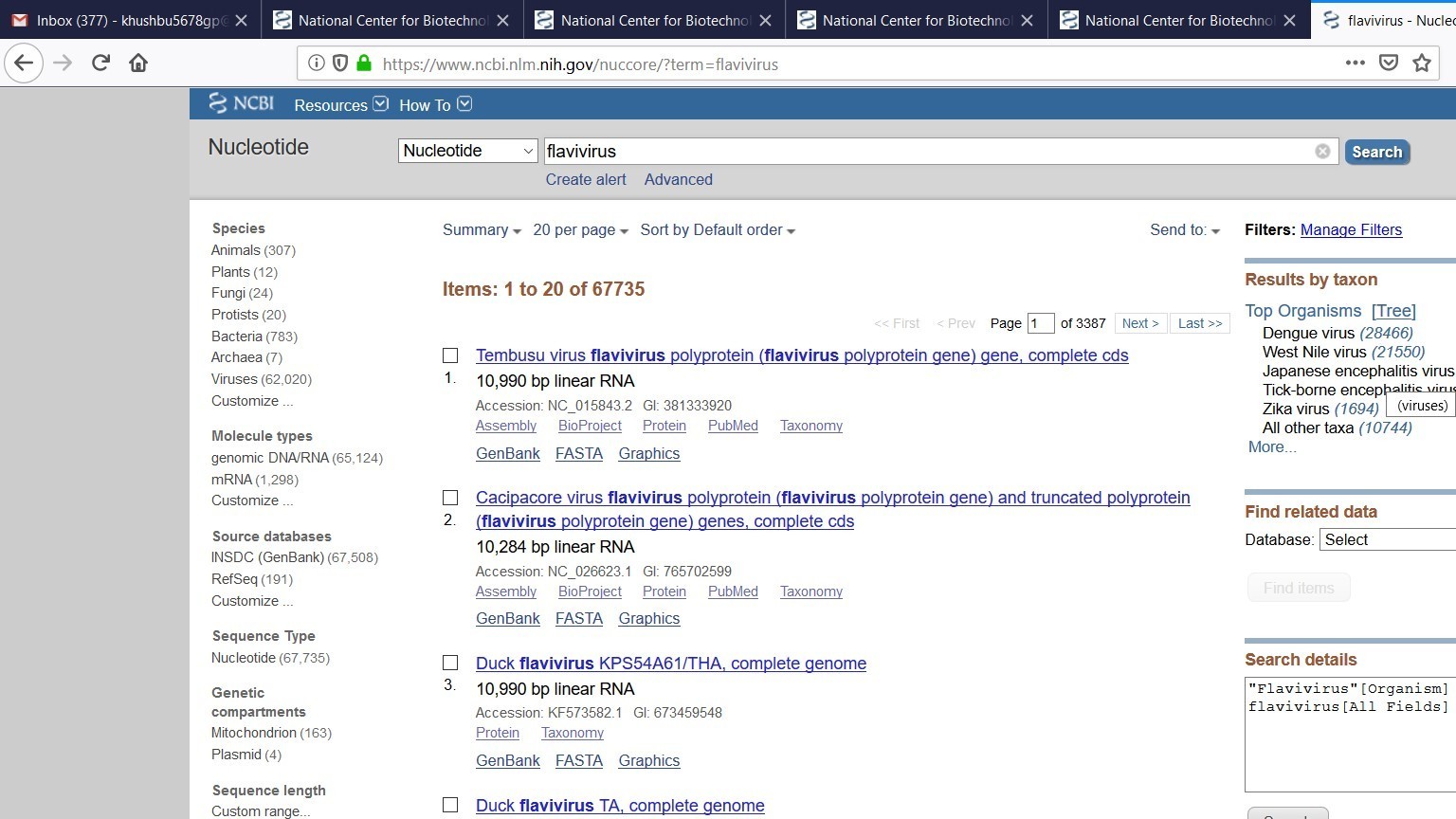
Steps:

Go to the National Center for Biotechnology Information Site https://[www.ncbi.nlm.nih.gov/](http://www.ncbi.nlm.nih.gov/)

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

Graphical user interface, text, application, email

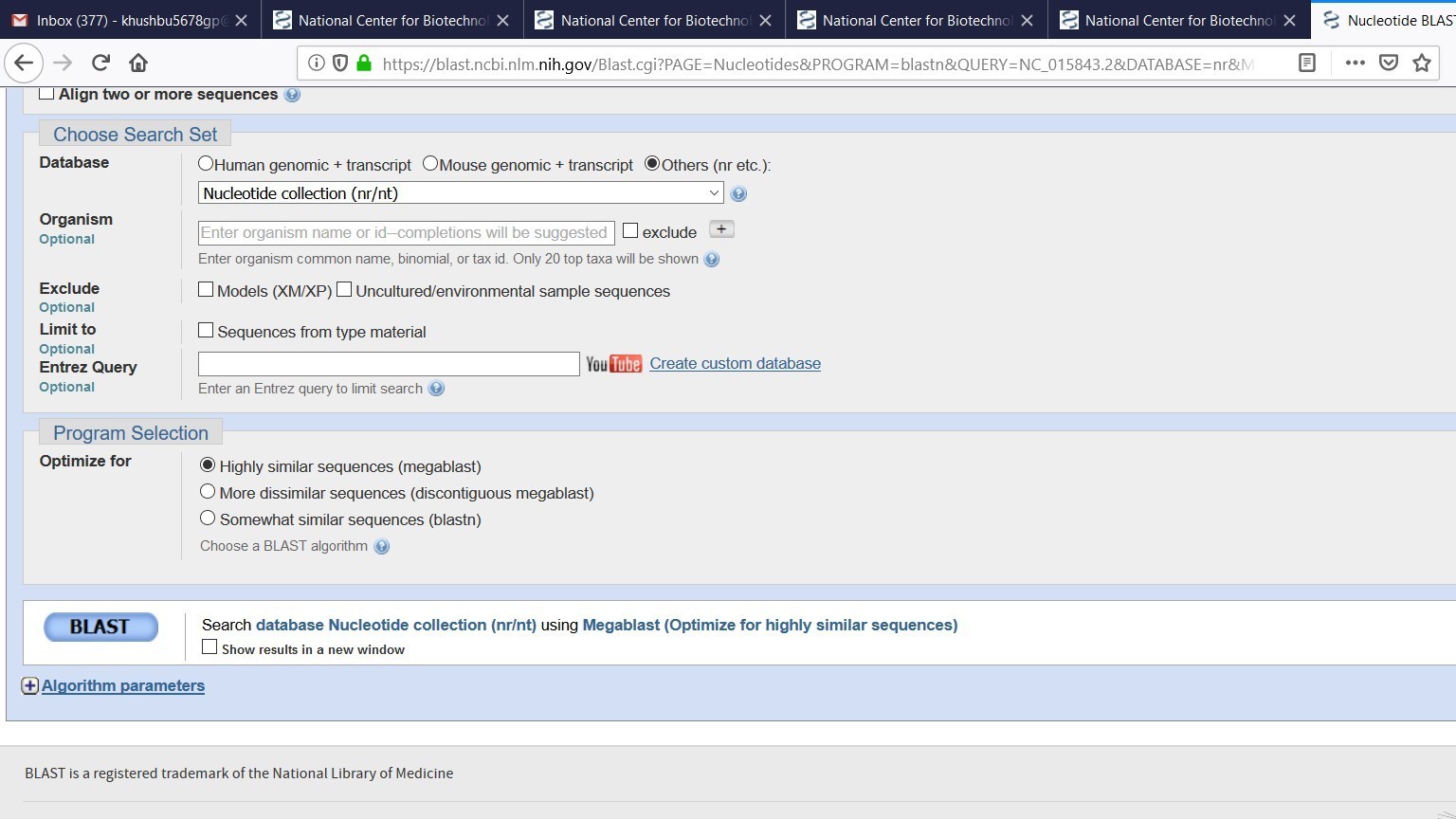
Description automatically generated



Graphical user interface, text, application, email

Description automatically generated

Run BLAST option we have to select

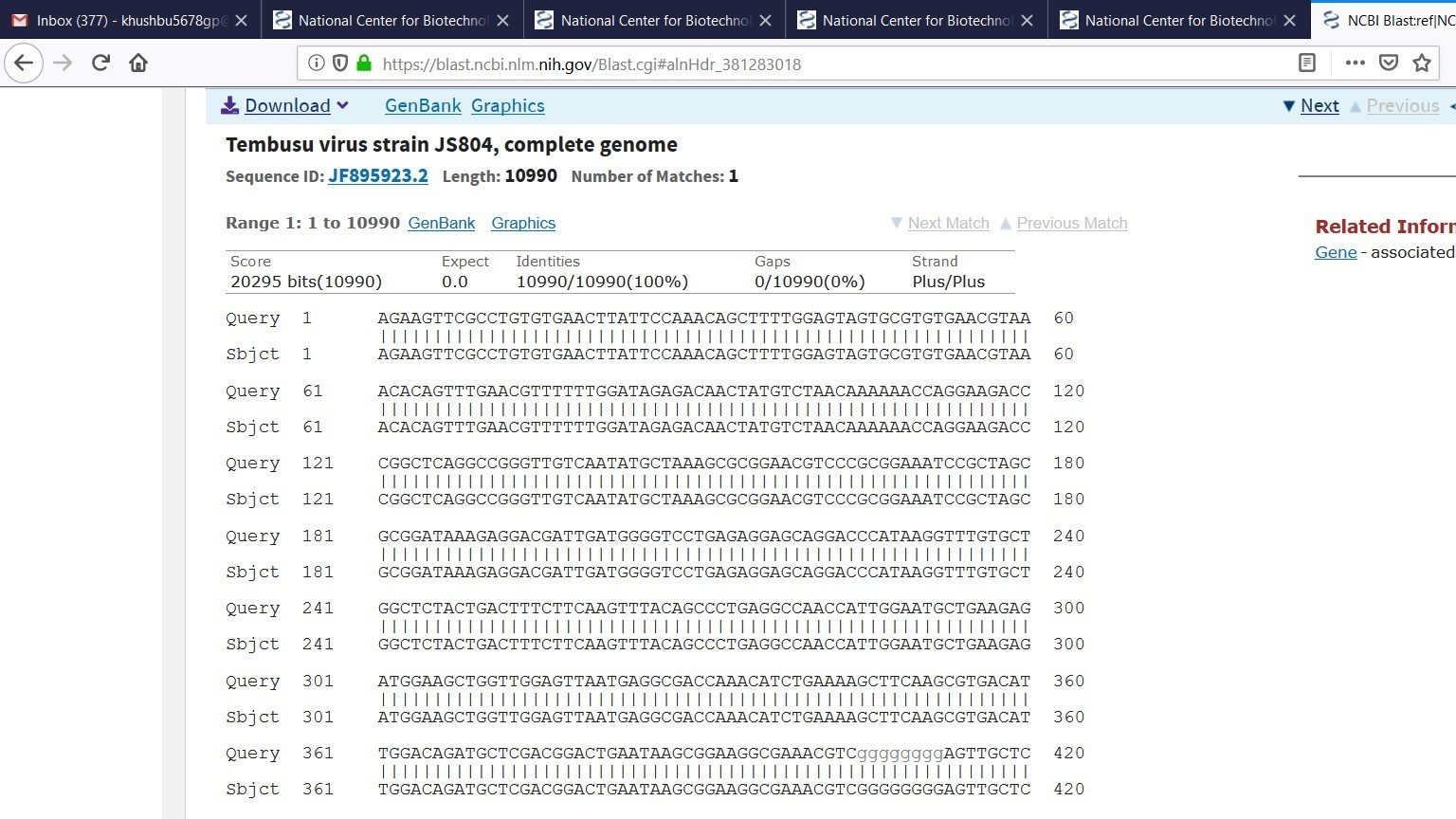


BLAST

Graphical user interface, text, application, email

Description automatically generated

Here the result will be display



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

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

Aim: Perform a BLAST search on any genes sequence and writer 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 |