**NAME: ASIF ERFAN KHAN**

**ROLL NUMBER: 546**

**COURSE: MSc CS**

**SUBJECT: BIOINFORMATICS**

**PRACTICAL: 1-10**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **INDEX** |  |  |
| **NO** | **DATE** | **TITLE** | **PAGE NO** | **SIGN** |
|  |  |  |  |  |
| 1 | 08-08-22 | Complementary DNA Sequence | 2 |  |
| 2 | 08-08-22 | Identity of Two protein sequence | 4 |  |
| 3 | 26-08-22 | Pairwise Sequence Alignment | 6 |  |
| 4 | 18-08-22 | Similarity between two protein sequence | 8 |  |
| 5 | 21-08-22 | Multiple Sequence Alignment | 10 |  |
| 6 |  | Motif Finding | 15 |  |
| 7 |  | Perform a BLAST search on any genes sequence and write a code to count the no of repetition of each nucleotide in the sequence | 16 |  |
| 8 | 29-09-22 | Regular Expression | 17 |  |
| 9 | 22-09-22 | Fingerprint | 18 |  |
| 10 |  | Retrieving 3D structure from PDB |  |  |

**PRACTICAL 1**

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

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)):

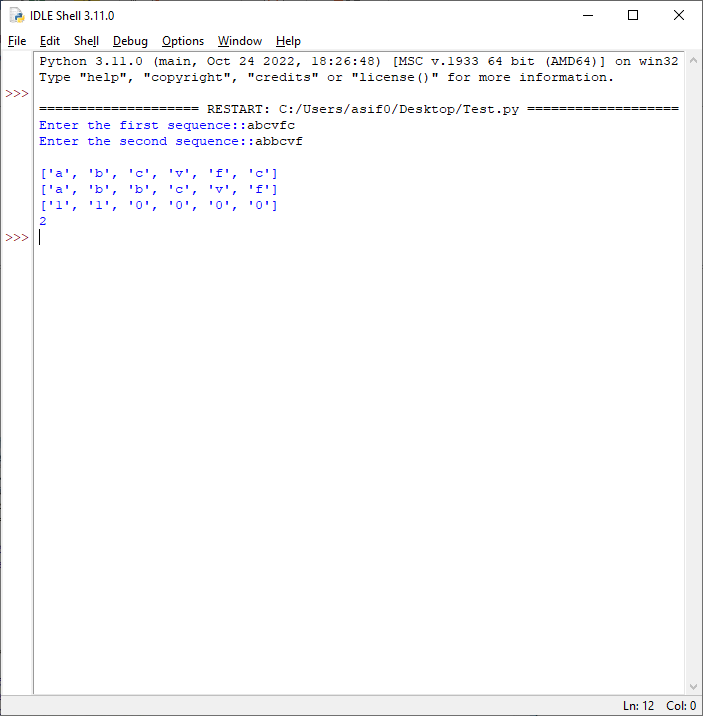
a.insert(k,'-')

else:

b.insert(k,'-')

return(a,b)

Pairwise\_alignment(seq1,seq2)

OUTPUT: 

**PRACTICAL 2**

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

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,'-')

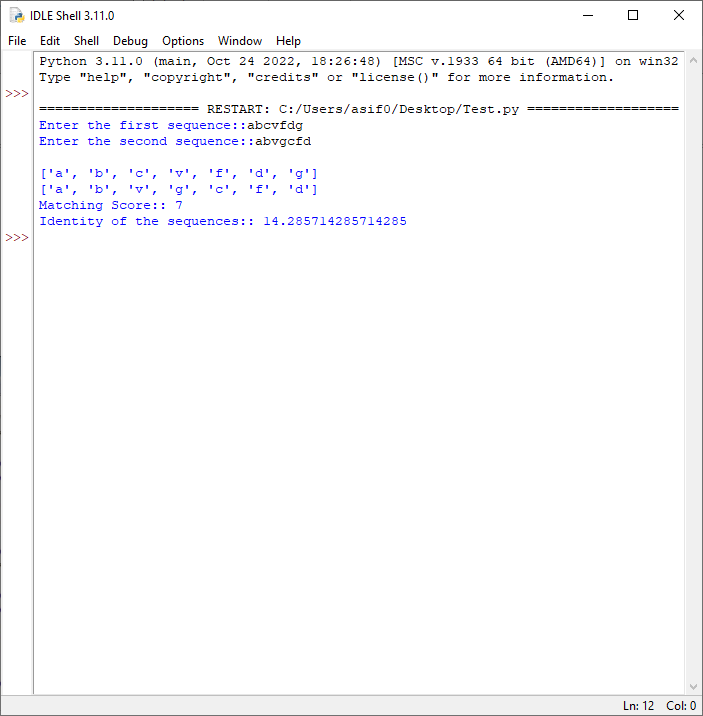
else:

b.insert(k,'-')

return(a,b)

find\_identity(seq1,seq2)

OUTPUT:



**PRACTICAL 3**

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

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

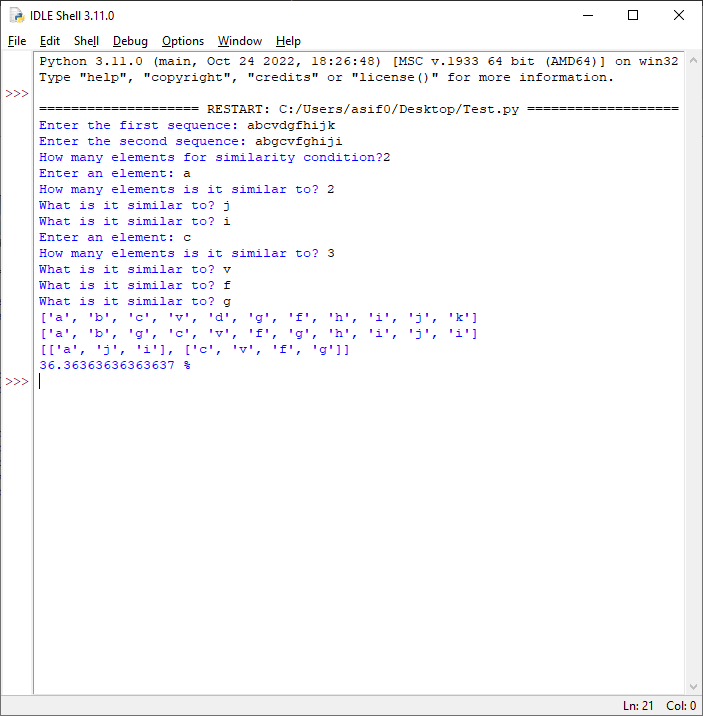
#calculating similarity

similarity= (score\*100)/len(o)

return similarity

print(compare(list(sequence\_one),list(sequence\_two),similarities),"%")

OUTPUT:



**PRACTICAL 4**

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

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

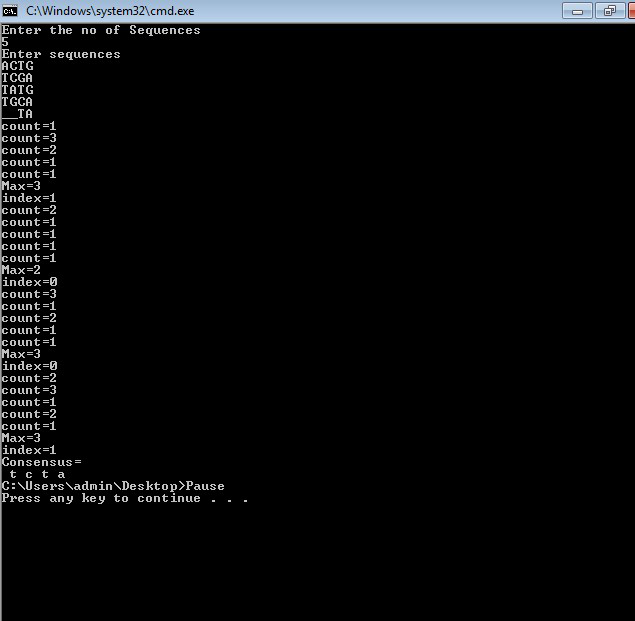
System.out.print(cons[j]);

}

}

}

OUTPUT:



**PRACTICAL 5**

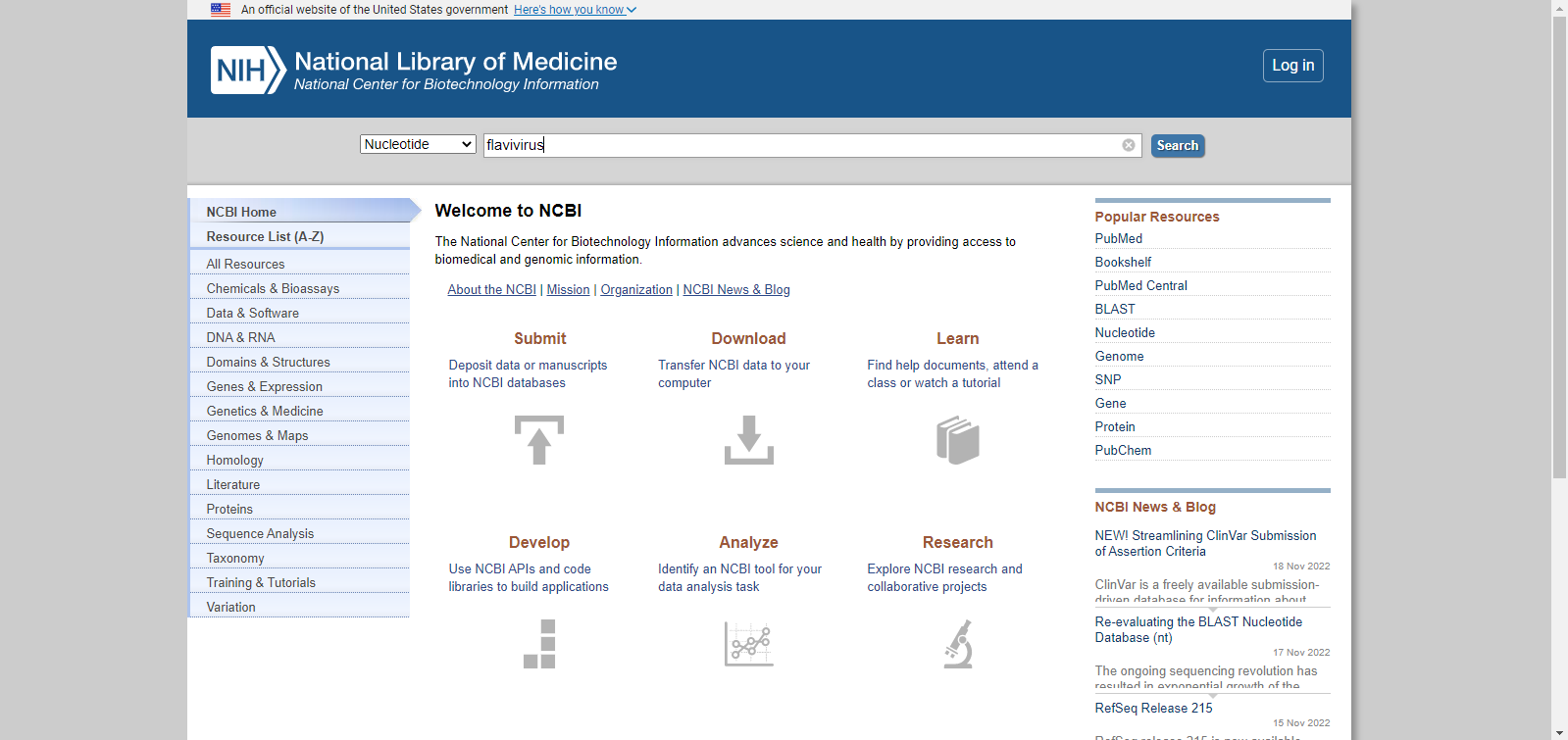
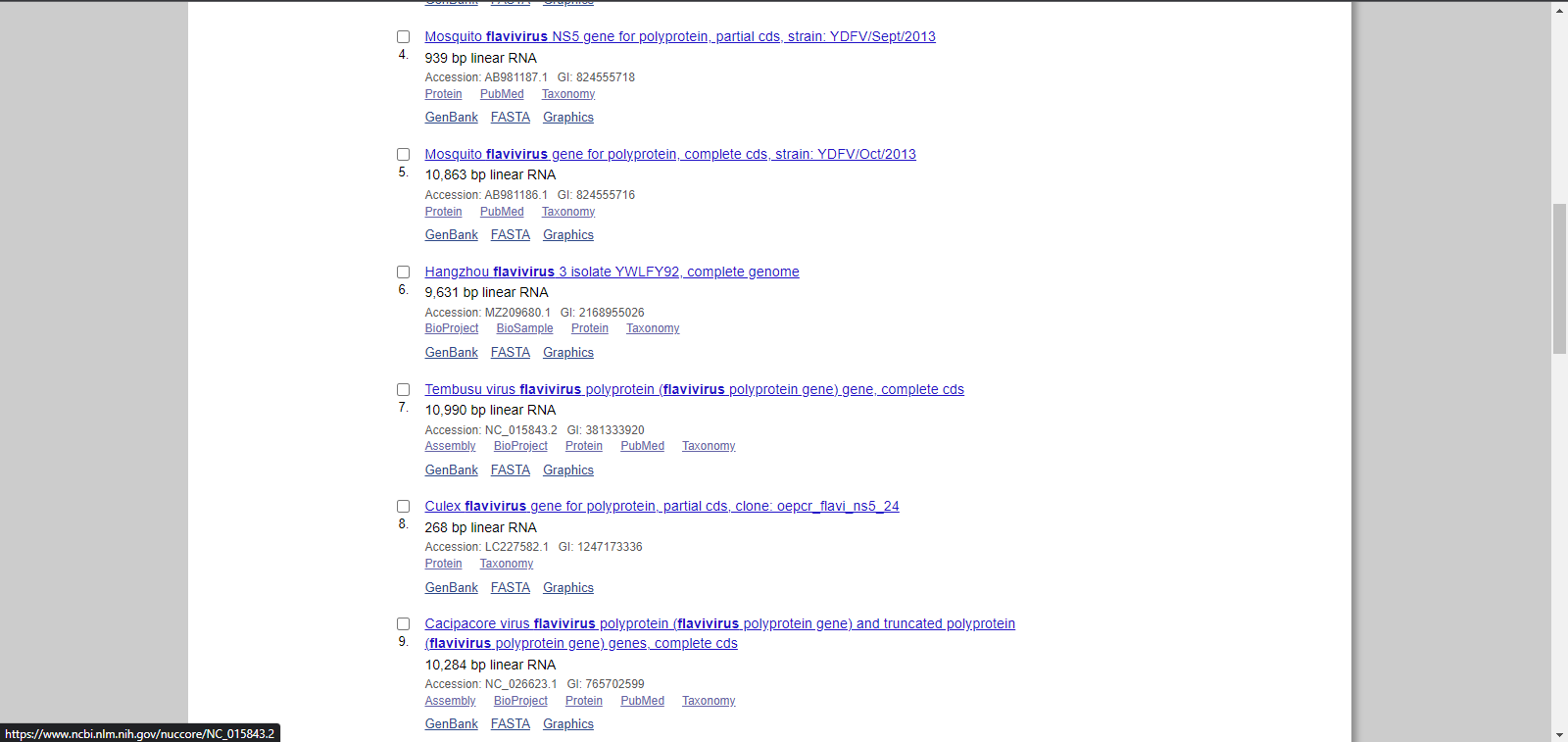
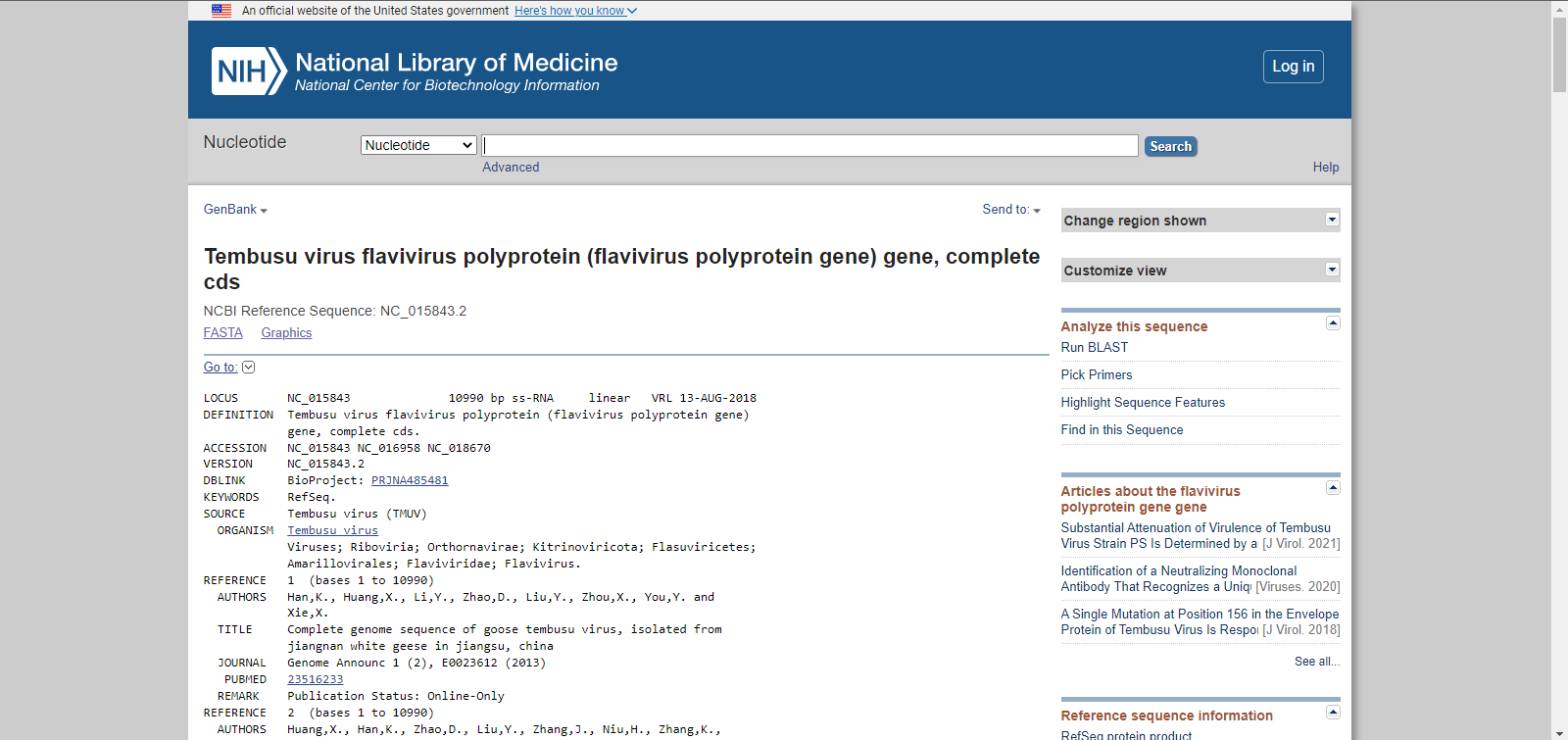
Aim: Perform a BLAST search on a specific gene sequence of a specify organism.

Steps:

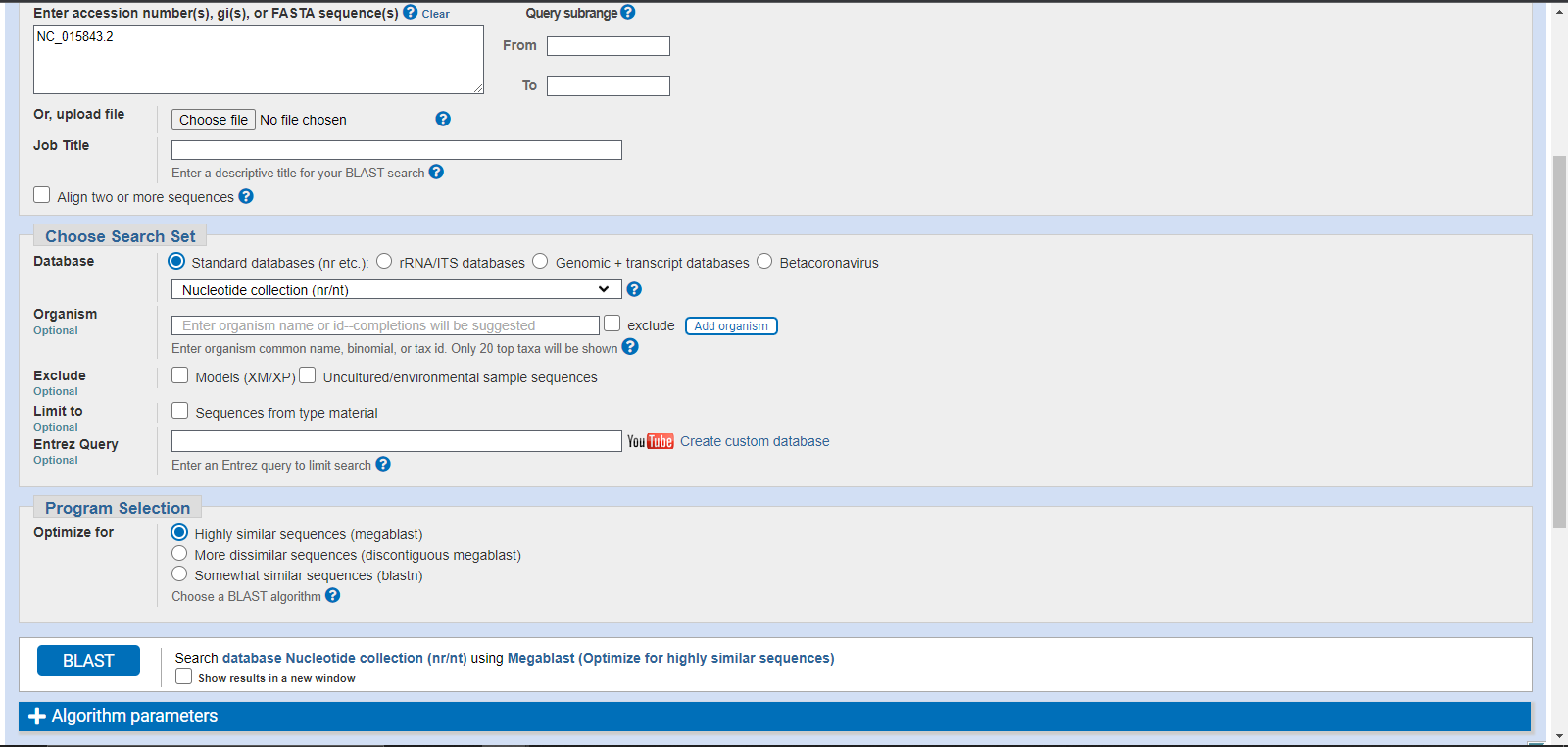
Go to the National Center for Biotechnology Information Site

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

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

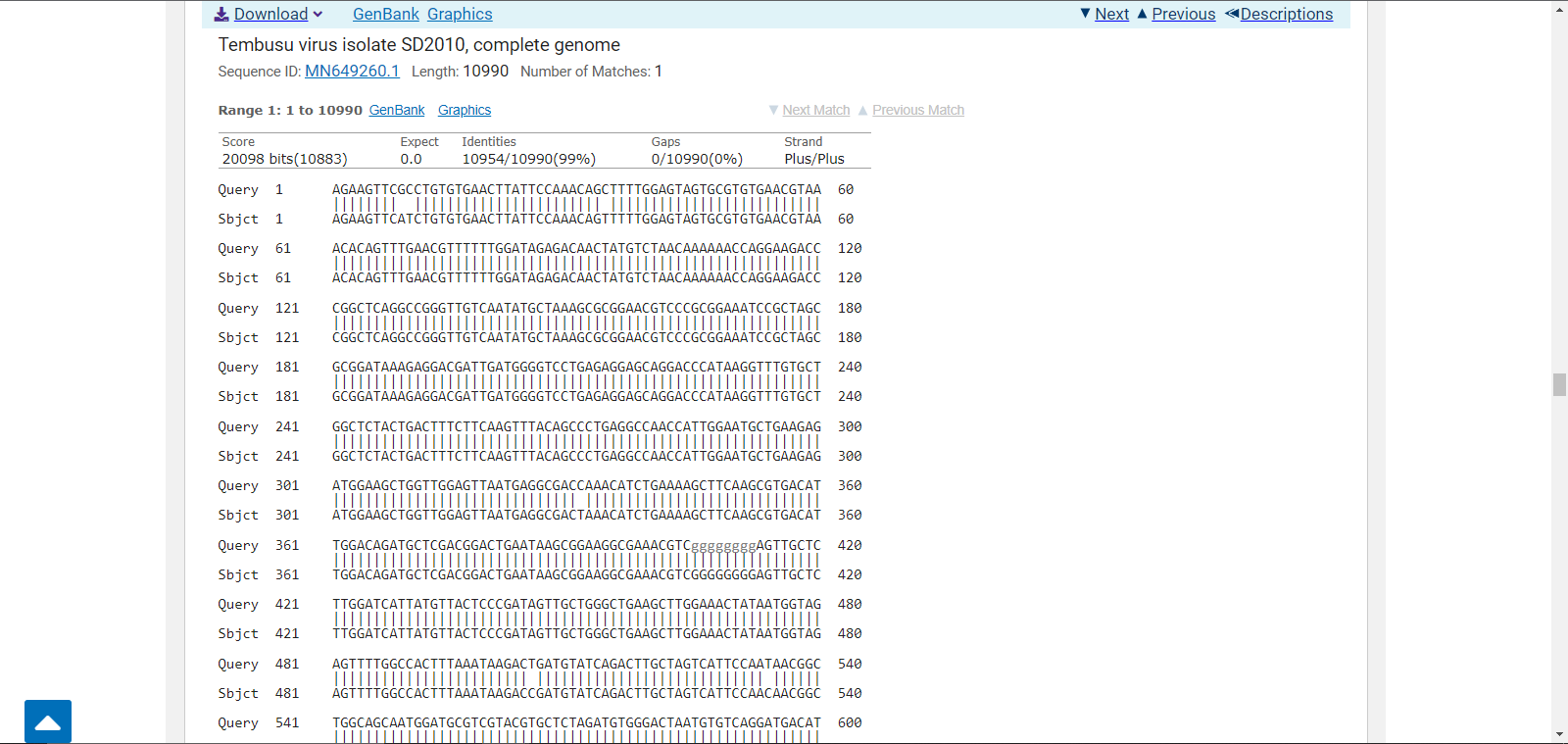
Run BLAST option we have to select



BLAST



Here the result will be display



**PRACTICAL 6**

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

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]

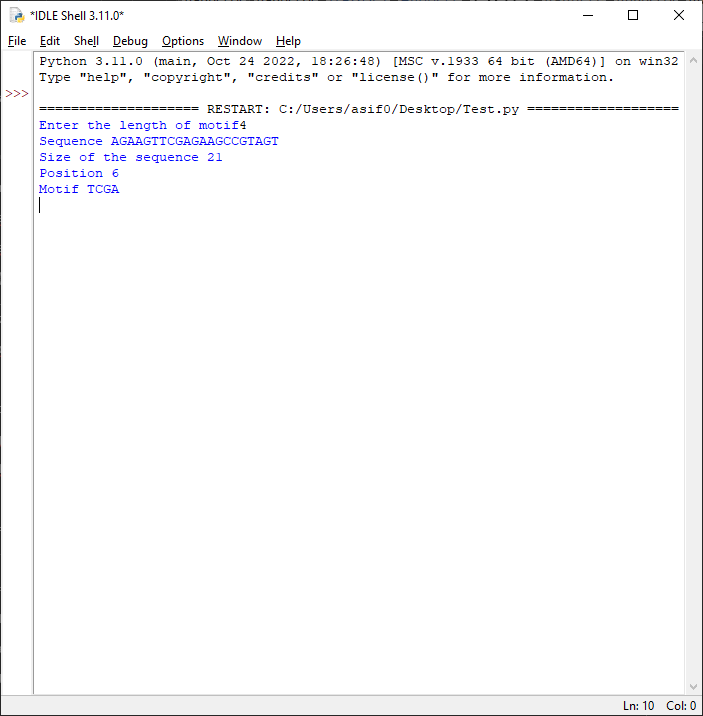
print("Match motif",str1)

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

file1.write(str1+" ")

i+=1

OUTPUT:



**PRACTICAL 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.

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

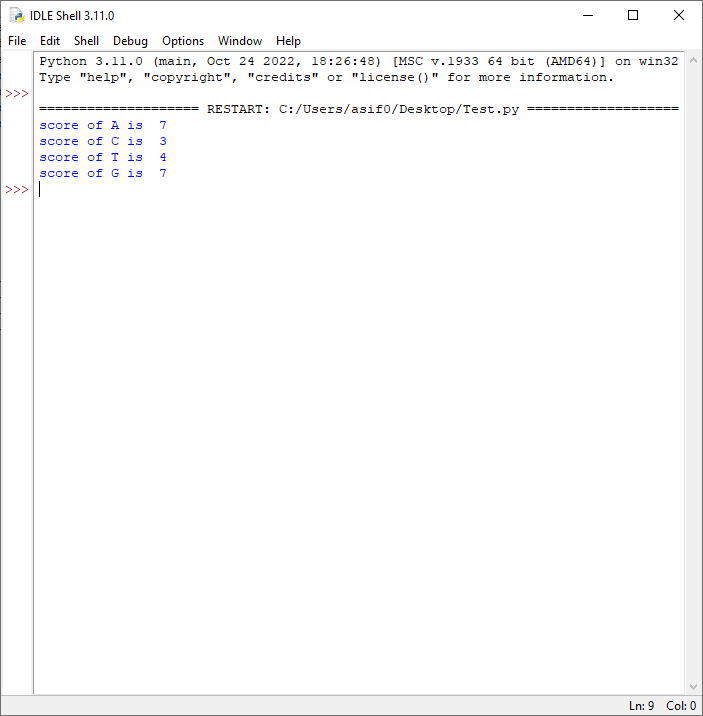
print("score of A is ",score\_A)

print("score of C is ",score\_C)

print("score of T is ",score\_T)

print("score of G is ",score\_G)

OUTPUT:



**PRACTICAL 8**

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

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")

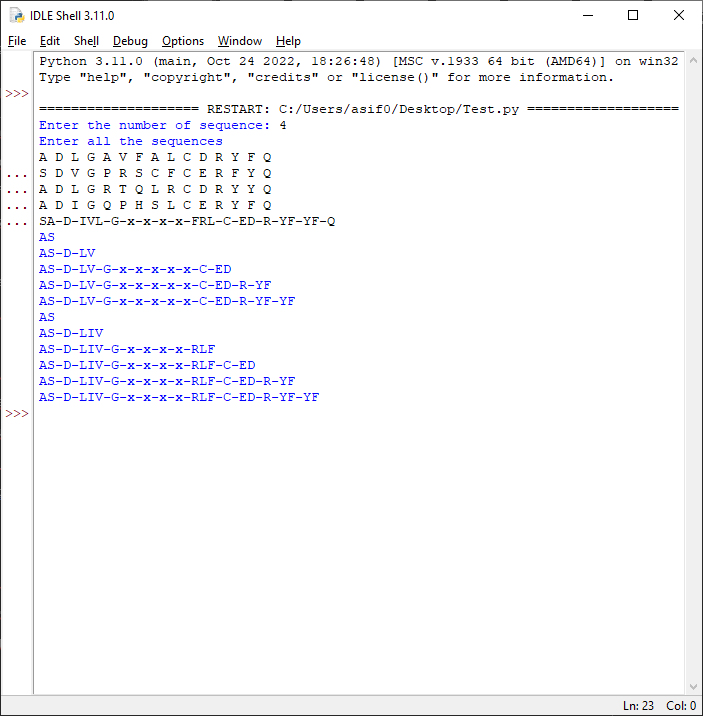
seq\_list=[]

for \_ in range(no\_of\_seq):

seq\_list.append(list(map(str, input("").split())))

gen\_reg\_exp(seq\_list, len(seq\_list[0]))

OUTPUT:



**PRACTICAL 9**

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

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: "))

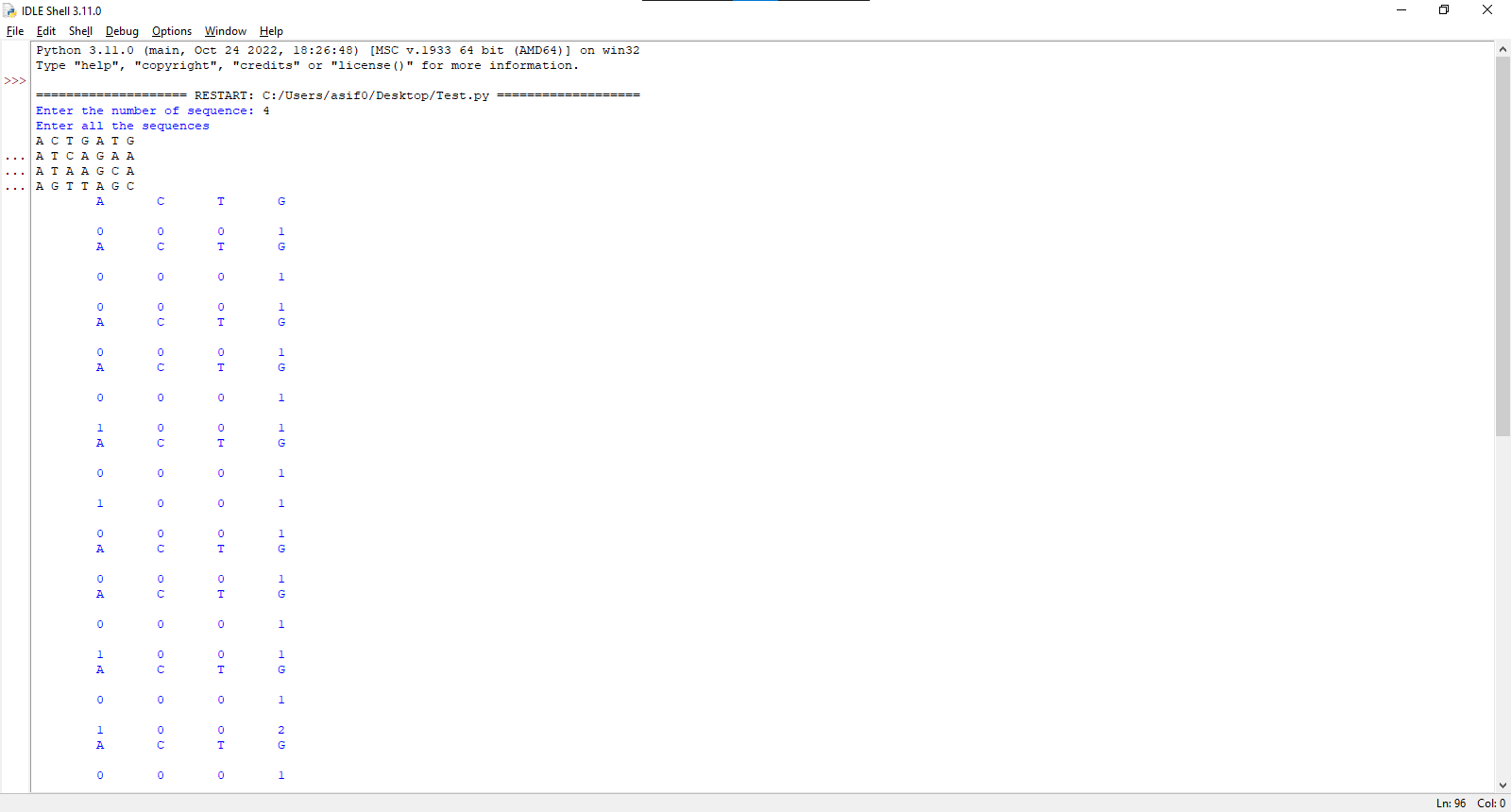
print("Enter all the sequences")

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: 

**PRACTICAL 10**

Aim: Retrieving 3D structure from PDB

To perform the current practical, you’ll be needing two things

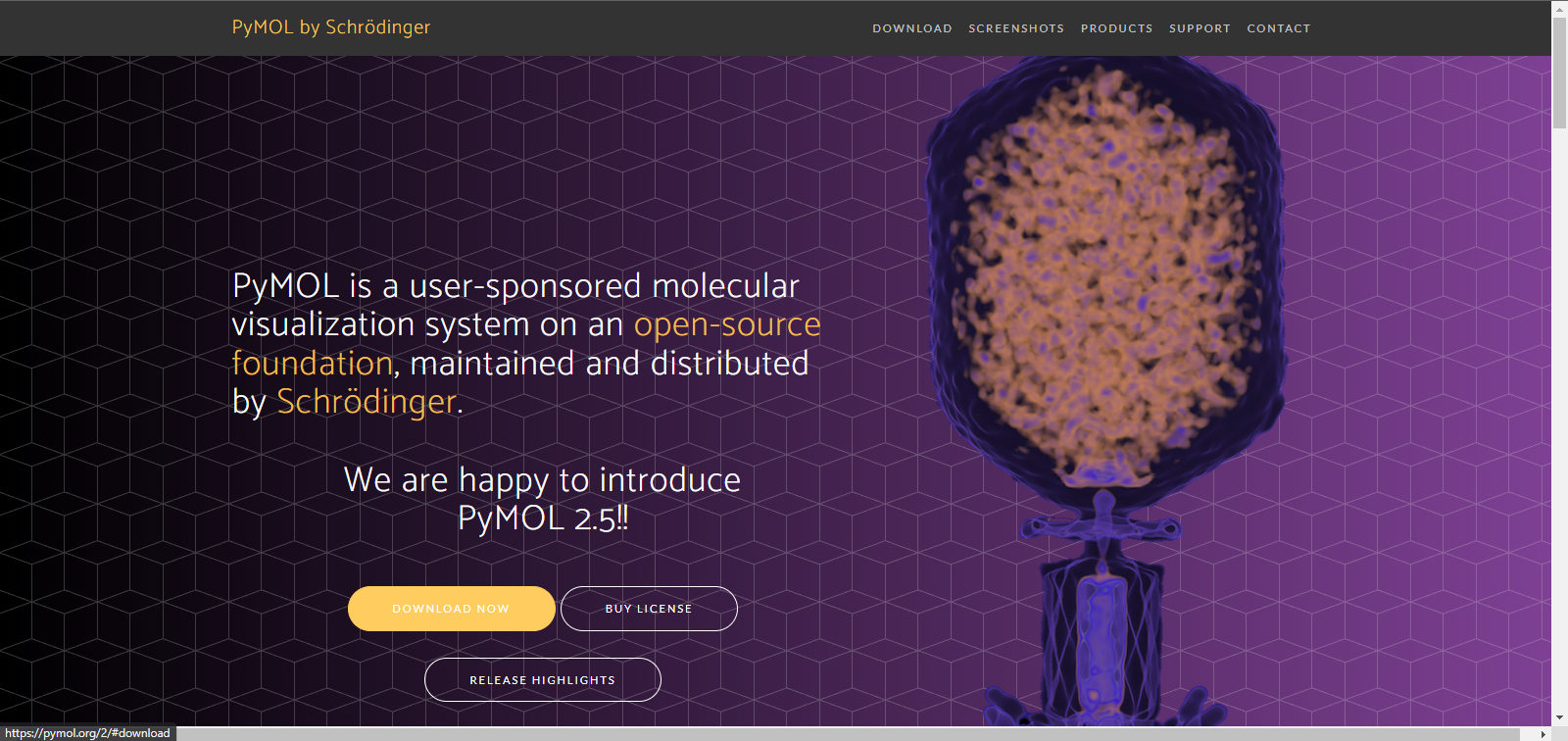
1. PyMOL (software)
2. Protein in .pdb format

**Installing PyMOL Software**

First, we need to install PyMOL. To do so open google and simply search PyMOL.

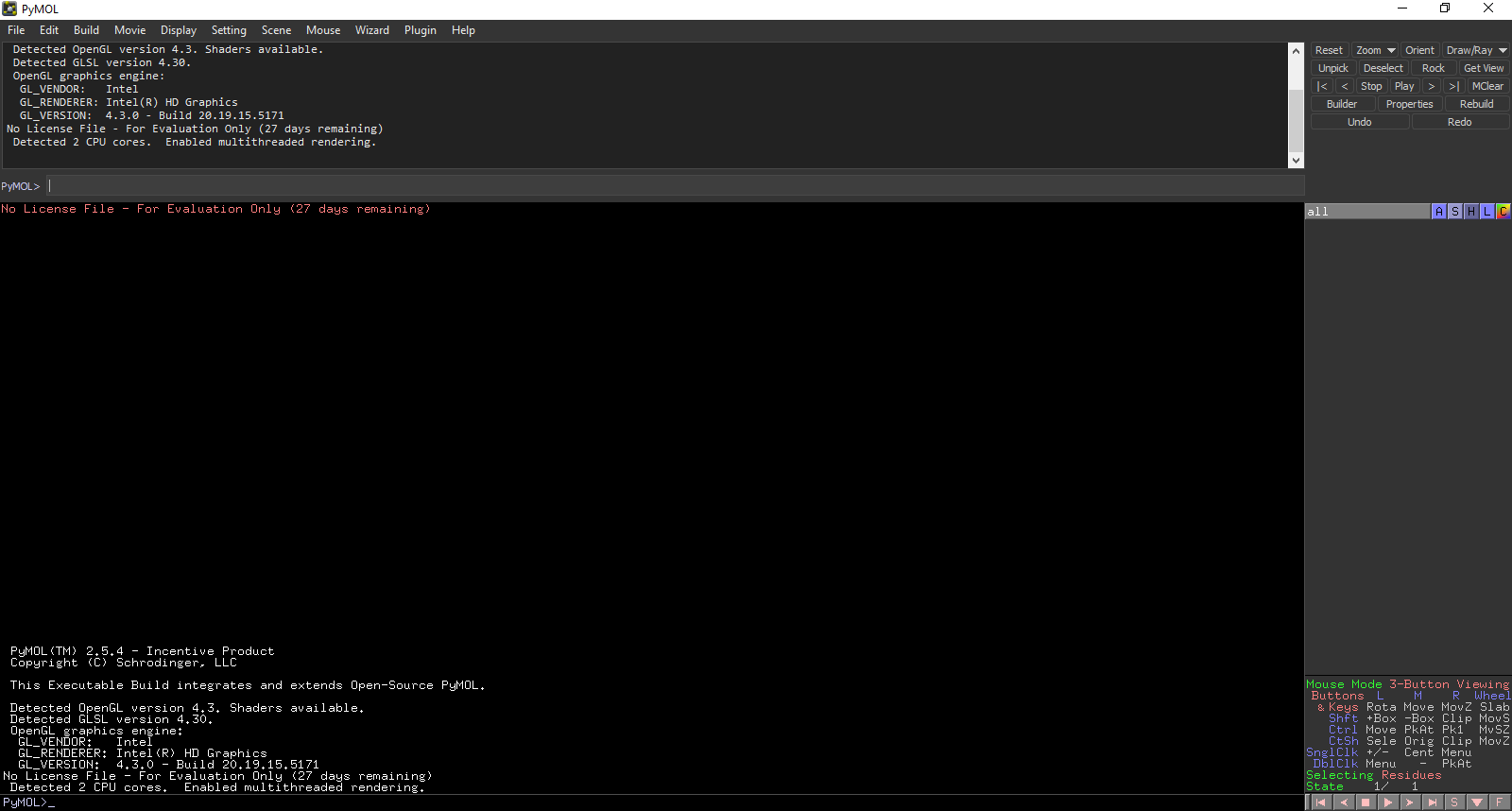


Open the first link and click Download Now.



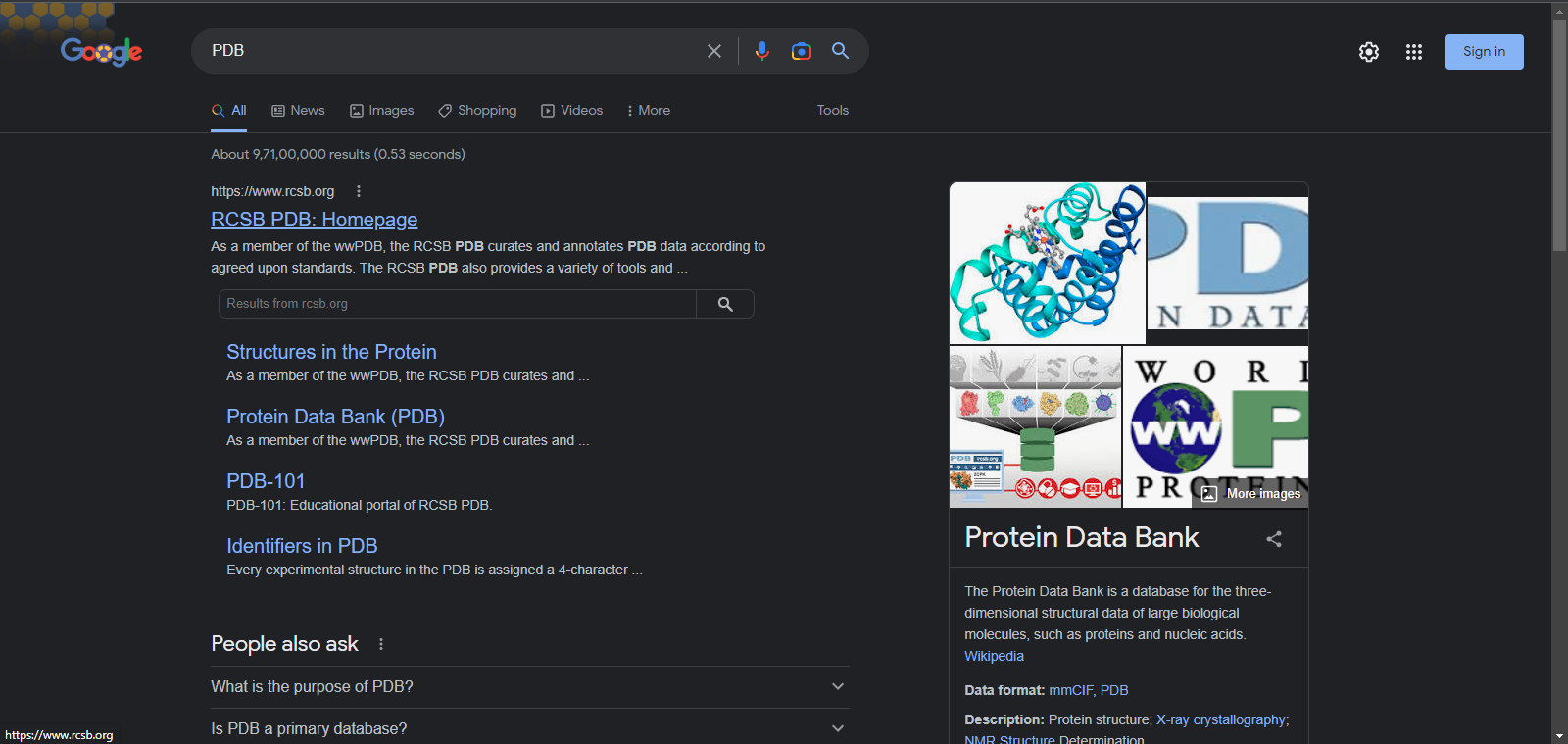
When done downloading install the software.

This is how the interface of the software looks like.



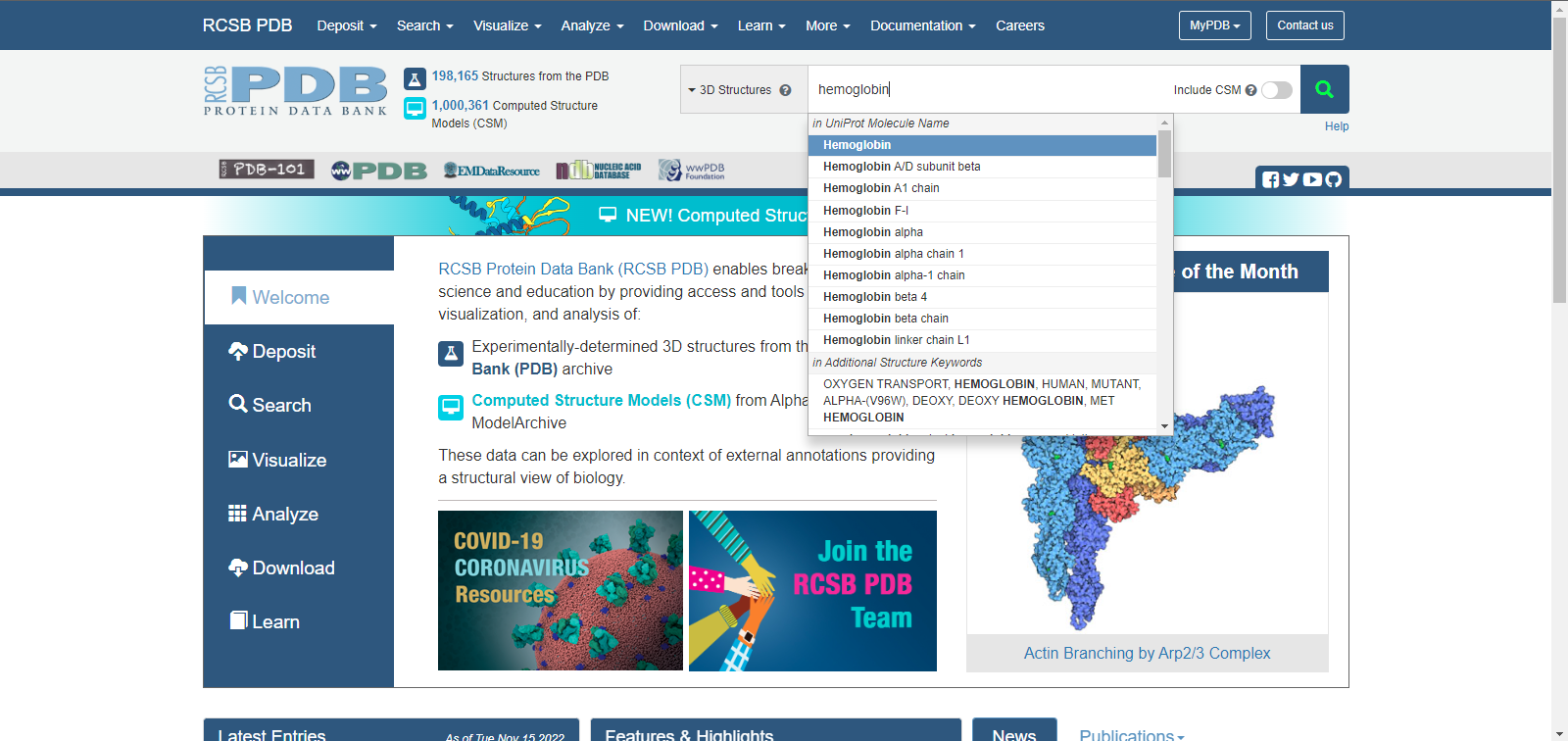
**Downloading Protein in .pdb format**

To download Protein open, google and search pdb or simply visit <https://www.rcsb.org/>

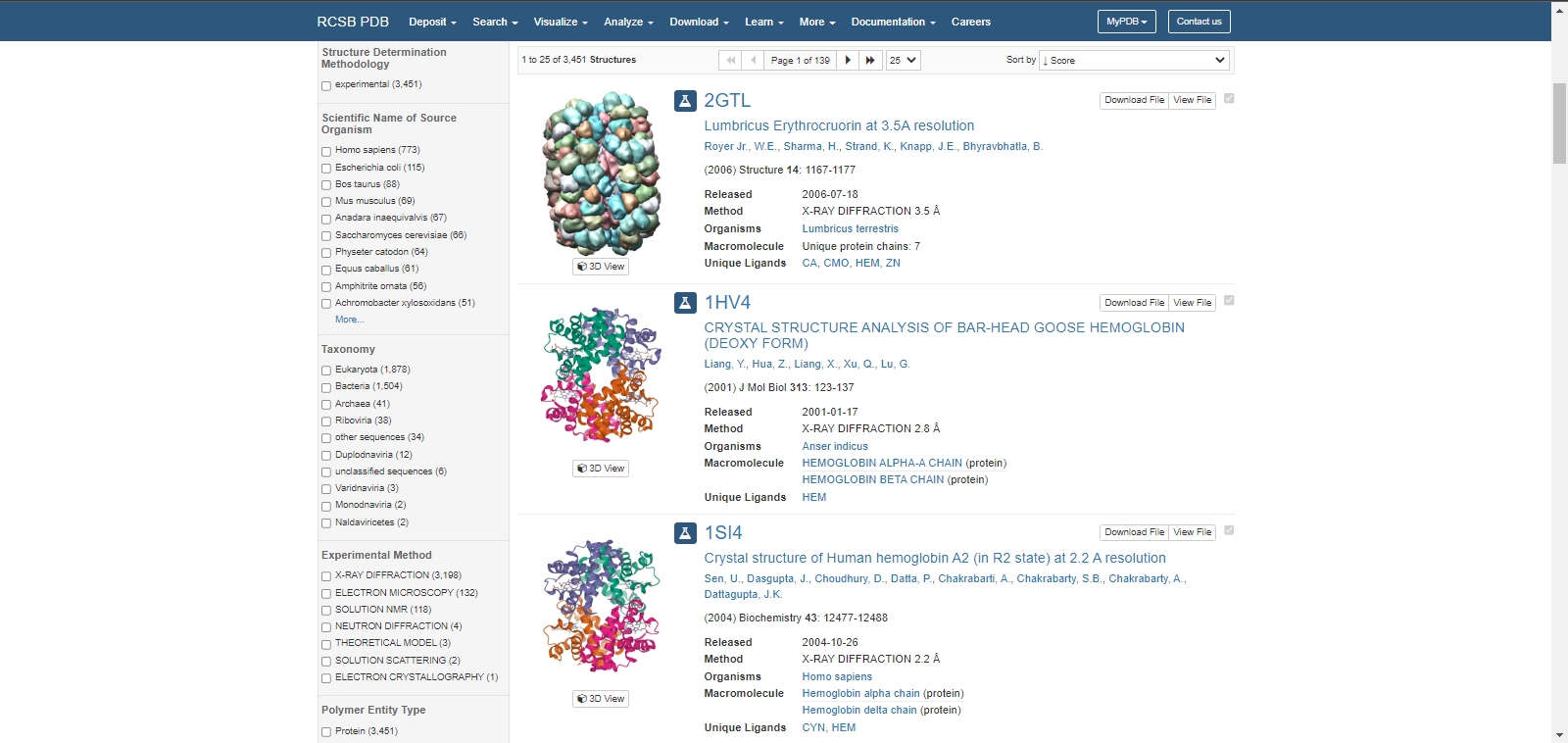


Open the first link named “RCSB PDB: Homepage”

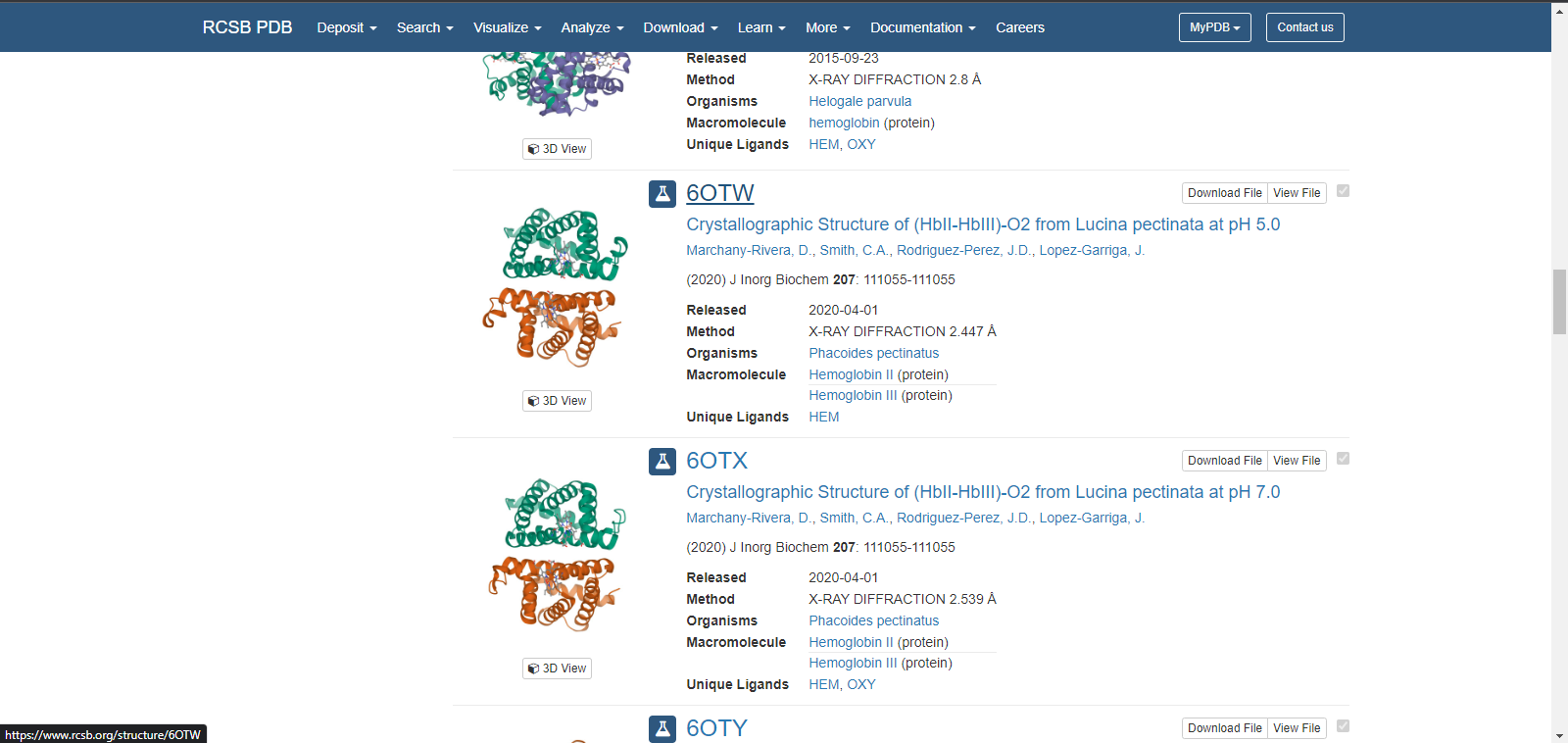
On the search bar search for “hemoglobin”



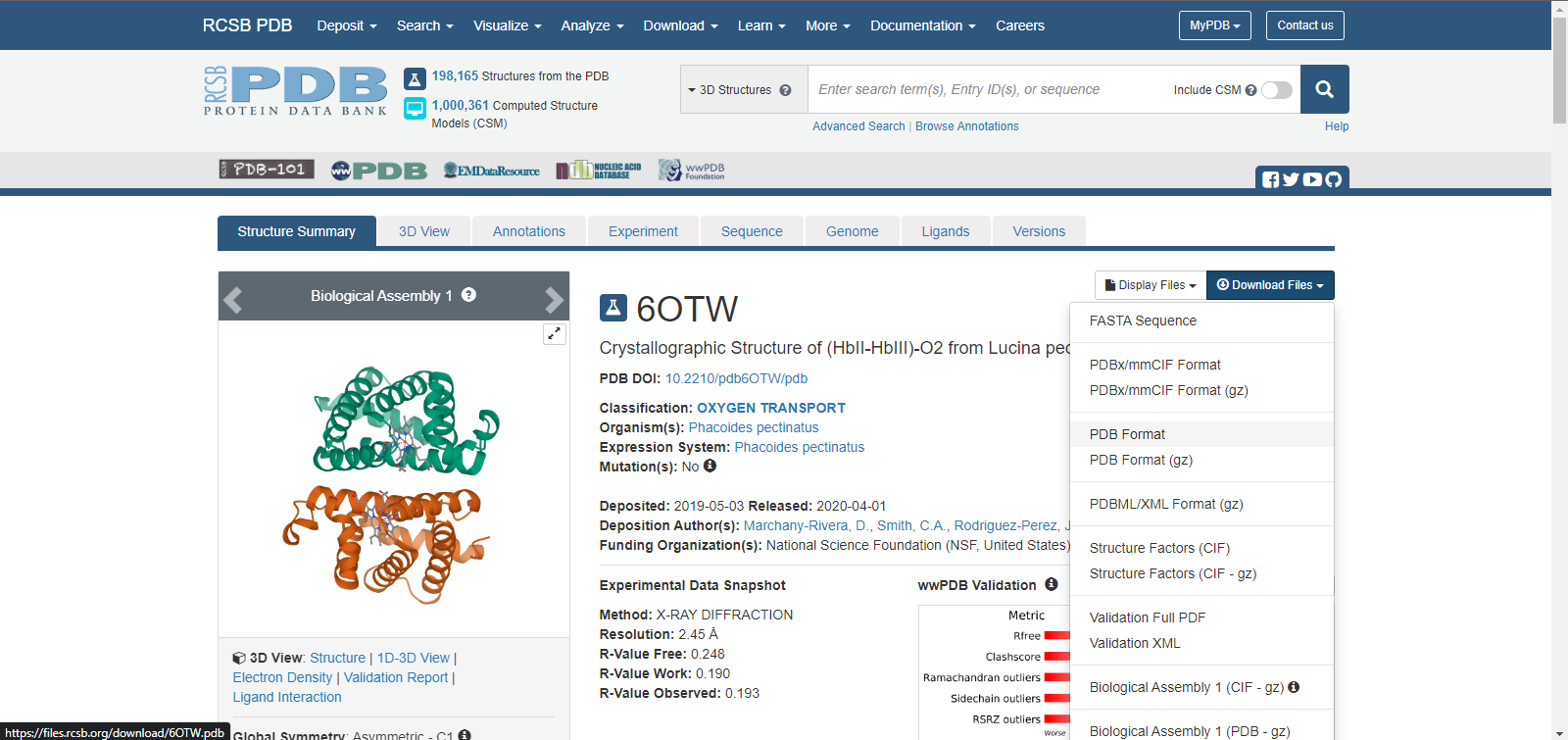
You’ll see all the proteins listed below like this.



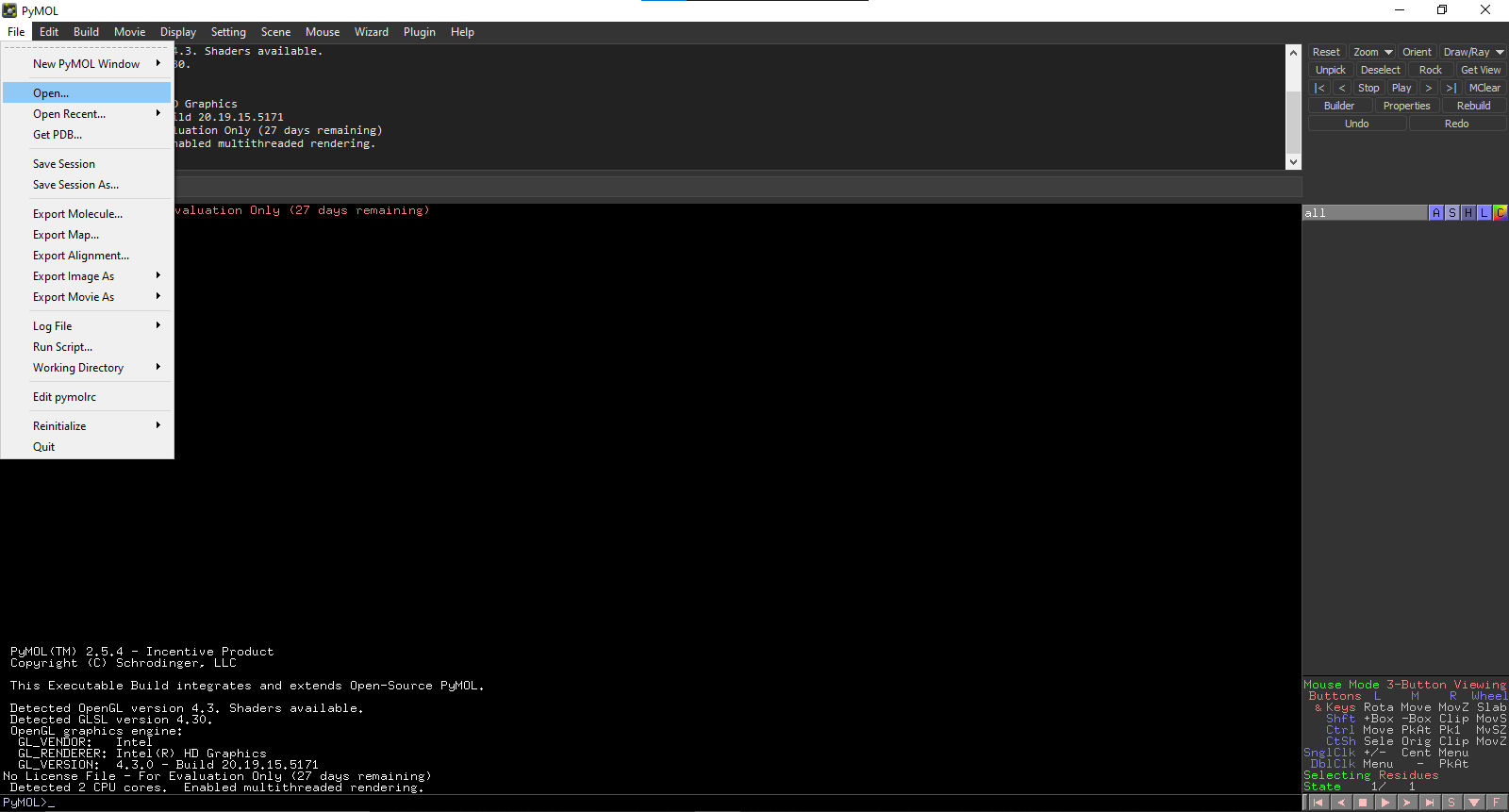
Scroll down until you find “4YU3” and open it or you can directly search for “6otw”



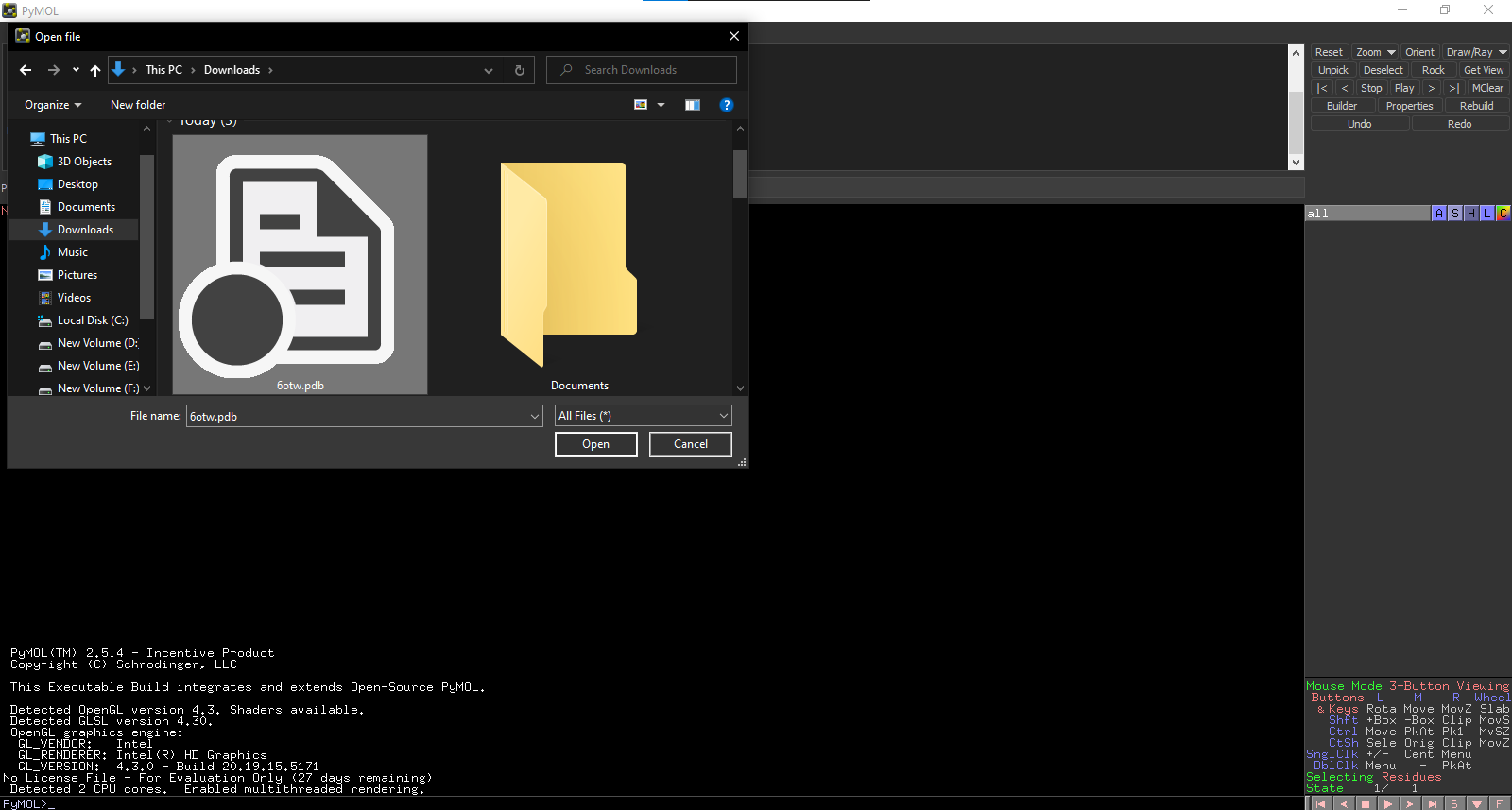
On the right-hand side, you’ll the download option. Click on it and download as PDB format.



Now that protein is downloaded open PyMOL and on top left corner click File > open



Now browse for the file you just downloaded (6otw.pdb)



This is how the screen of imported file looks like…

