**PROGRAM 1**

**AIM**

**To write a program to count mononucleotide frequencies.**

**SOURCE CODE**

#START

#!/usr/bin/perl

print "Program to calculate mononucleotide frequency\n\n";

print "Enter a DNA sequence:\n";

$seq = <STDIN>;

chomp($seq);

$seq = uc($seq);

$seq = reverse $seq;

$length = length($seq);

$i = 0;

@arrSeq = ();

#can also use split function

#For checking invalid characters

while ( $i < $length ) {

$temp = chop($seq);

if ( $temp eq 'A' || $temp eq 'T' || $temp eq 'G' || $temp eq 'C' ) {

push( @arrSeq, $temp );

$i++;

}

else {

print "You have entered invalid sequence\n";

goto end;

}

}

print "\nYou have entered:\n@arrSeq \n";

#calculation of nucleotides

$a = 0;

$t = 0;

$g = 0;

$c = 0;

foreach $char (@arrSeq) {

$a++ if ( $char eq 'A' );

$t++ if ( $char eq 'T' );

$g++ if ( $char eq 'G' );

$c++ if ( $char eq 'C' );

}

print "\n";

print( "Frequency of A is $a of $length i.e ", ( $a / $length ) \* 100, "%\n" );

print( "Frequency of T is $t of $length i.e ", ( $t / $length ) \* 100, "%\n" );

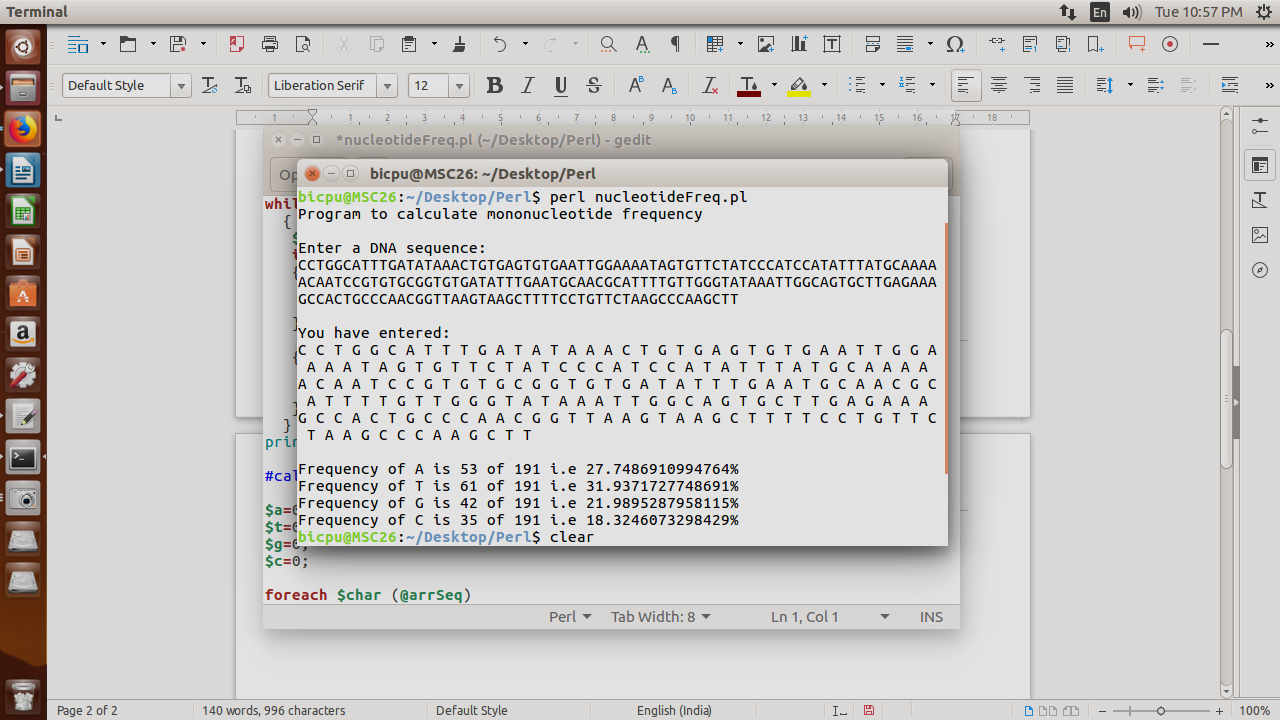
print( "Frequency of G is $g of $length i.e ", ( $g / $length ) \* 100, "%\n" );

print( "Frequency of C is $c of $length i.e ", ( $c / $length ) \* 100, "%\n" );

end:

#END

**OUTPUT**



**PROGRAM 2**

**AIM**

**To write a program to count dinucleotide frequencies.**

**SOURCE CODE**

#START

#!/usr/bin/perl

print "Program to calculate dinucleotide frequency\n\n";

print "Enter a DNA sequence:\n";

$input = <STDIN>;

chomp($input);

$input = uc($input);

$len = length($input);

$i = 0;

@arr = split( '', $input );

print "\nYou have entered: \n@arr\n";

#calculation of frequency

while ( $i < $len ) {

$aa++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'AA' );

$at++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'AT' );

$ag++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'AG' );

$ac++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'AC' );

$ta++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'TA' );

$tt++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'TT' );

$tg++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'TG' );

$tc++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'TC' );

$ga++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'GA' );

$gt++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'GT' );

$gg++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'GG' );

$gc++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'GC' );

$ca++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'CA' );

$ct++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'CT' );

$cg++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'CG' );

$cc++ if ( $arr[$i] . $arr[ $i + 1 ] eq 'CC' );

$i++;

}

print "\n";

print ("Frequency of AA is $aa \n") if ( $aa != 0 );

print ("Frequency of AT is $at \n") if ( $at != 0 );

print ("Frequency of AG is $ag \n") if ( $ag != 0 );

print ("Frequency of AC is $ac \n") if ( $ac != 0 );

print ("Frequency of TA is $ta \n") if ( $ta != 0 );

print ("Frequency of TT is $tt \n") if ( $tt != 0 );

print ("Frequency of TG is $tg \n") if ( $tg != 0 );

print ("Frequency of TC is $tc \n") if ( $tc != 0 );

print ("Frequency of GA is $ga \n") if ( $ga != 0 );

print ("Frequency of GT is $gt \n") if ( $gt != 0 );

print ("Frequency of GG is $gg \n") if ( $gg != 0 );

print ("Frequency of GC is $gc \n") if ( $gc != 0 );

print ("Frequency of CA is $ca \n") if ( $ca != 0 );

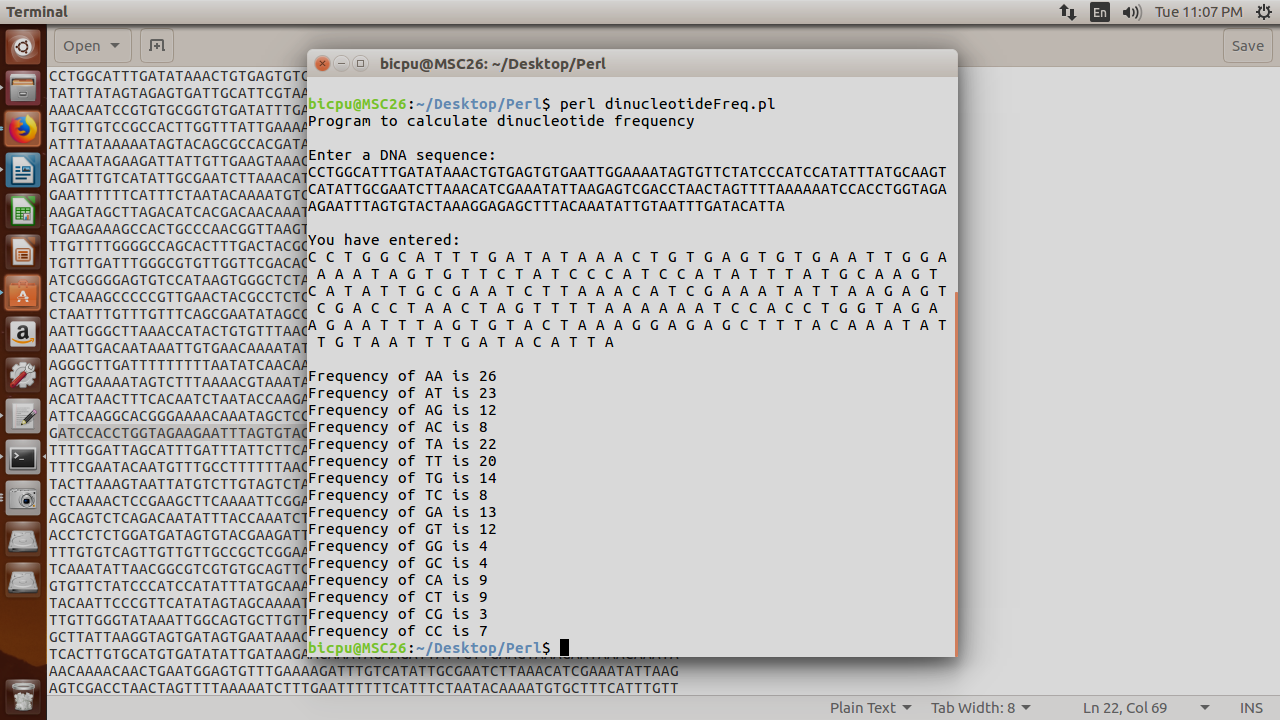
print ("Frequency of CT is $ct \n") if ( $ct != 0 );

print ("Frequency of CG is $cg \n") if ( $cg != 0 );

print ("Frequency of CC is $cc \n") if ( $cc != 0 );

#END

**OUTPUT**



**PROGRAM 3**

**AIM**

**To write a program to translate a DNA sequence to a protein sequence.**

**SOURCE CODE**

#START

#!/usr/bin/perl

$file1 = "sequence1.txt";

open( IN, "<", $file1 ) || die("Unable to open $file1\n$!");

@dna1 = <IN>;

$len = @dna1;

$dna = "";

for ( $i = 1 ; $i < $len ; $i++ ) {

$dna .= $dna1[$i];

}

%codon = (

"ATT" => I, "ATC" => I, "ATA" => I, "CTT" => L, "CTC" => L, "CTA" => L,

"CTG" => L, "TTA" => L, "TTG" => L, "GTT" => V, "GTC" => V, "GTA" => V,

"GTG" => V, "TTT" => F, "TTC" => F, "ATG" => M, "TGT" => C, "TGC" => C,

"GCT" => A, "GCC" => A, "GCA" => A, "GCG" => A, "GGT" => G,

"GGC" => G, "GGA" => G, "GGG" => G, "CCT" => P, "CCC" => P,

"CCA" => P, "CCG" => P, "ACT" => T, "ACC" => T, "ACA" => T, "ACG" => T,

"TCT" => S, "TCC" => S, "TCA" => S, "TCG" => S, "AGT" => S, "AGC" => S,

"TAT" => Y, "TAC" => Y, "TGG" => W, "CAA" => Q, "CAG" => Q,

"AAT" => N, "AAC" => N, "CAT" => H, "CAC" => H, "GAA" => E,

"GAG" => E, "GAT" => D, "GAC" => D, "AAA" => K, "AAG" => K,

"CGT" => R, "CGC" => R, "CGA" => R, "CGG" => R, "AGA" => R,

"AGG" => R, "TAA" => "stop", "TAG" => "stop", "TGA" => "stop");

$protein = translateDNA($dna);

print "\nThe protein sequence is: \n\n$protein\n";

sub translateDNA() {

my @input = @\_;

my $dna = join( "", @input );

chomp($dna);

my $len = length($dna);

my $prot = "";

print "DNA sequence is:\n\n$dna";

for ( $i = 0 ; $i < $len - 2 ; $i++ ) {

my $su = substr( $dna, $i, 3 );

$prot .= $codon{$su};

$i += 2;

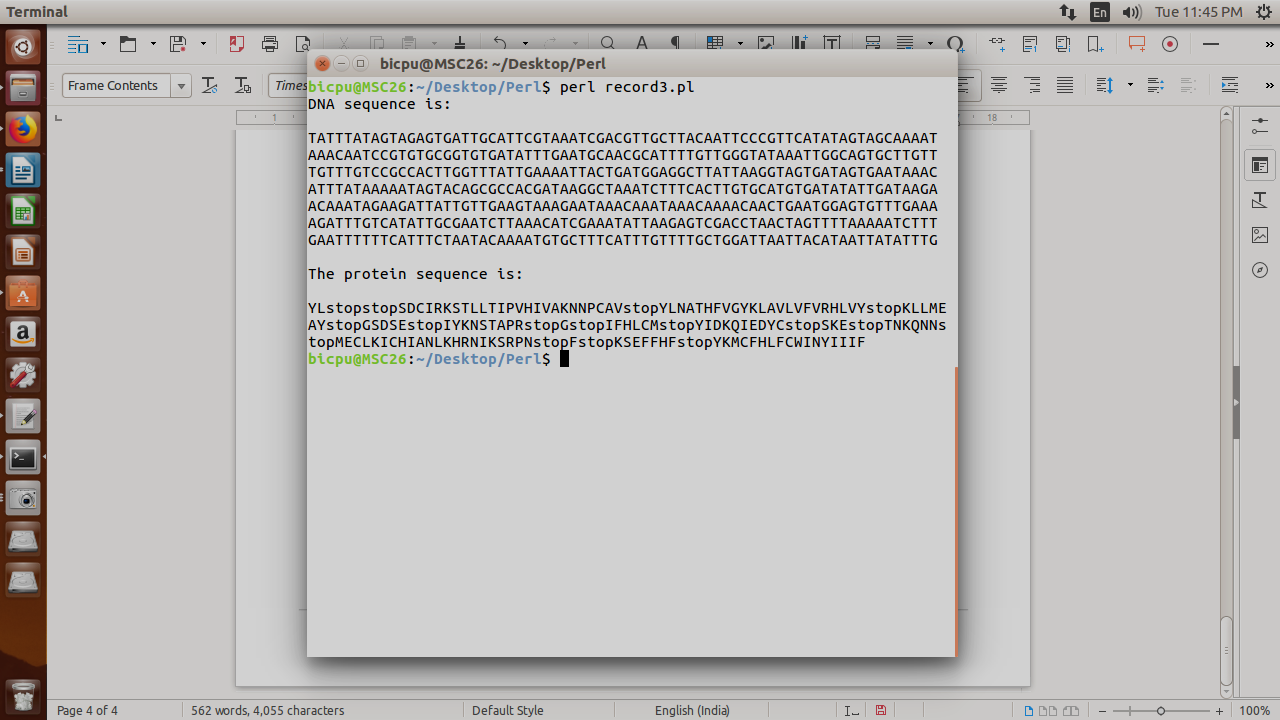
}

return $prot;

}

#END

**OUTPUT**



**PROGRAM 4**

**AIM**

**To write a program to calculate molecular weight of a protein sequence.**

**SOURCE CODE**

#START

#!/usr/bin/perl

print "\n\t\t\t\t\*\*\*\*WELCOME\*\*\*\*\n";

print "This program calculates the total moleculat weight of your protein sequence\n\n";

again:

up:

print "Press '1' for giving input from keyboard.\nPress '2' for giving an input file.\n";

$opn = (<STDIN>);

chomp $opn;

if ($opn eq '1')

{

print "\nENTER YOUR PROTEIN SEQUENCE:\n";

$input = <STDIN>;

chomp $input;

$input = uc $input;

}

elsif ($opn eq '2')

{

print "\nENTER YOUR FILE NAME:\n";

$name = <STDIN>;

chomp $name;

open (IN,"<$name") or (die $!);

$temp ='';

@lines = ();

#to read FASTA files with skipped first line

while(<IN>)

{

chomp $\_;

push (@lines,$\_);

}

shift (@lines);

$input = join('',@lines);

$input = uc $input;

}

else

{

print "\n!!! oops again !!!\n\n";

goto up;

}

print "\nCalculating....\n";

sleep (2);

%molwt =

('A',89.1,'R',174.2,'N',132.1,'D',133.1,'C',121.2,'E',147.1,'Q',146.2,'G',75.1,'H',155.2,'I',131.2,

'L',131.2,'K',146.2,'M',149.2,'F',165.2,'P',115.1,'S',105.1,'T',119.1,'W',204.2,'Y',181.2,'V',117.1);

@seq = split('',$input);

unless ($input =~ /([^ARNDCEQGHILKMFPSTWYV])/)

{

$totalwt = 0;

foreach (@seq)

{

$totalwt += $molwt{$\_};

}

print "\nTotal molecular is weight is $totalwt\n";

}

else

{

print "Bad protein sequence with '$1', Try another protein\n\n";

repeat:

print "Do you wish to try again??(Y/N)\n";

$opt = <STDIN>;

chomp $opt;

$opt = uc $opt;

if ($opt eq 'Y'){print "\n";goto again;}

if ($opt eq 'N'){print "Thankyou for using :)\n";goto end;}

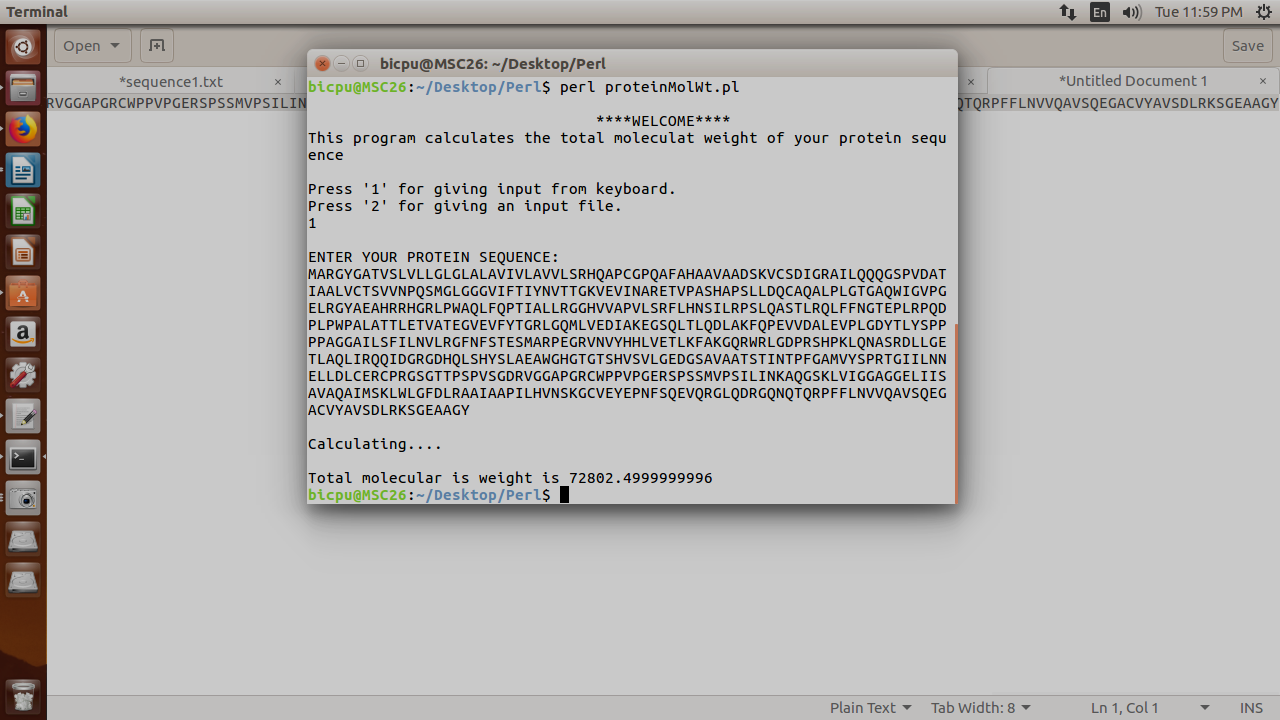
else {print "\n!!!oops again!!!\n\n";goto repeat;}

}

end:

#END

**OUTPUT**



**PROGRAM 5**

**AIM**

**To write a Perl script to list the contents of the current directory and use file tests to examine whether each file is readable/ writable/ executable and also display the file size.**

**SOURCE CODE**

#START

#!/usr/bin/perl

opendir DIR, "." or die "Could not open current directory: $!";

print "\nThe current directory contains: \n";

while ($\_ = readdir(DIR)) {

next if $\_ eq "." or $\_ eq "..";

print $\_, ":", " ", "\t";

print "\t";

print "d" if -d $\_;

print "r" if -r \_;

print "w" if -w \_;

print "e" if -e \_;

print "\t";

print -s \_ if -r \_ and -f \_ ;

print "\n";

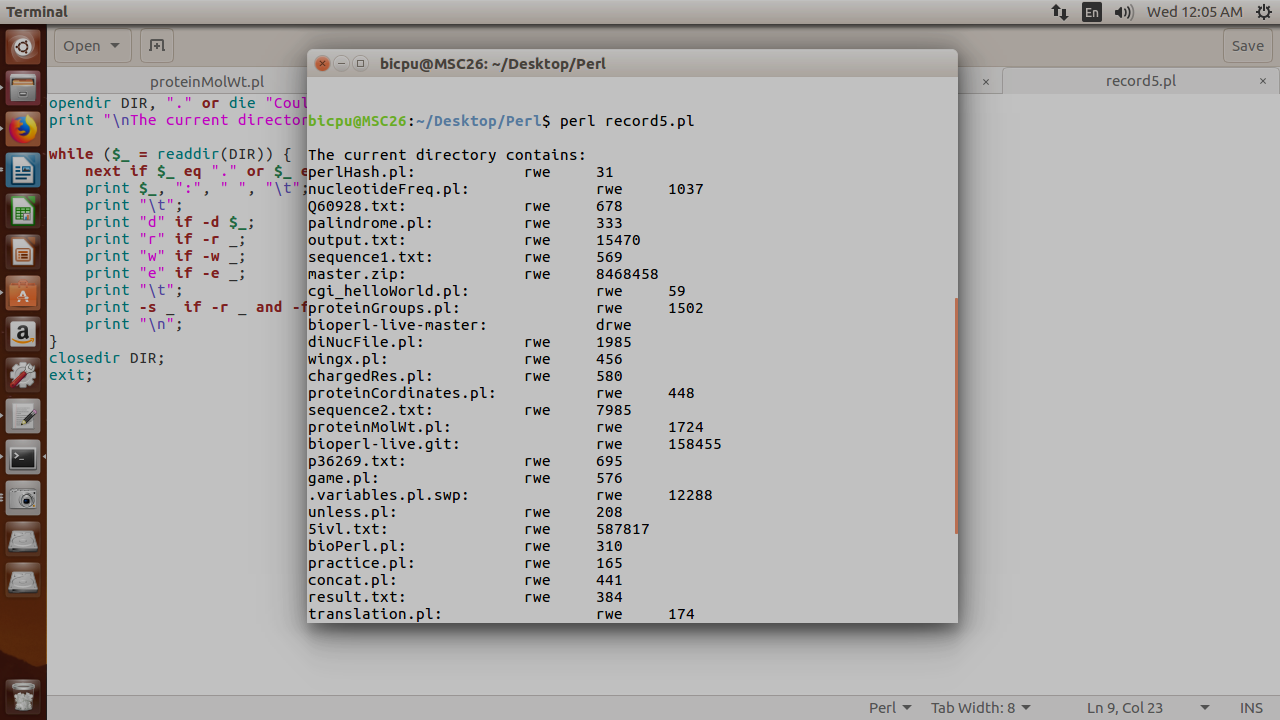
}

closedir DIR;

exit;

#END

**OUTPUT**



**PROGRAM 6**

**AIM**

**To write a CGI script to show your personal details.**

**SOURCE CODE**

**HTML CODE**

<html>

<head>

<title>Tell us something about you</title>

</head>

<body>

<h1>Welcome</h1>

<p>Please tell us about you</p>

<form action='/cgi-bin/Ankur/aboutProg.pl' method="get">

<p>Name: <input type="text" name="name" size="30"></p>

</tr>

<table>

<tr>

<td>Age: </td>

<td>

<select name="age" size="1">

<option>below15</option>

<option>16 - 25</option>

<option>26 - 35</option>

<option>36 - 45</option>

<option>above45</option>

</select>

</td>

<tr>

<td>Gender: </td>

<td><input type="radio" name="gender" value="male">Male

<input type="radio" name="gender" value="female">Female

</td>

</tr>

<tr>

<td>Hobbies</td>

<td><input type="checkbox" name="hobby" value="Sports">Sports

<input type="checkbox" name="hobby" value="Music">Music

<input type="checkbox" name="hobby" value="Reading">Reading

<input type="checkbox" name="hobby" value="Singing">Singing

</td>

</tr>

<tr>

<td></td>

<td><input type="submit"></td>

</tr>

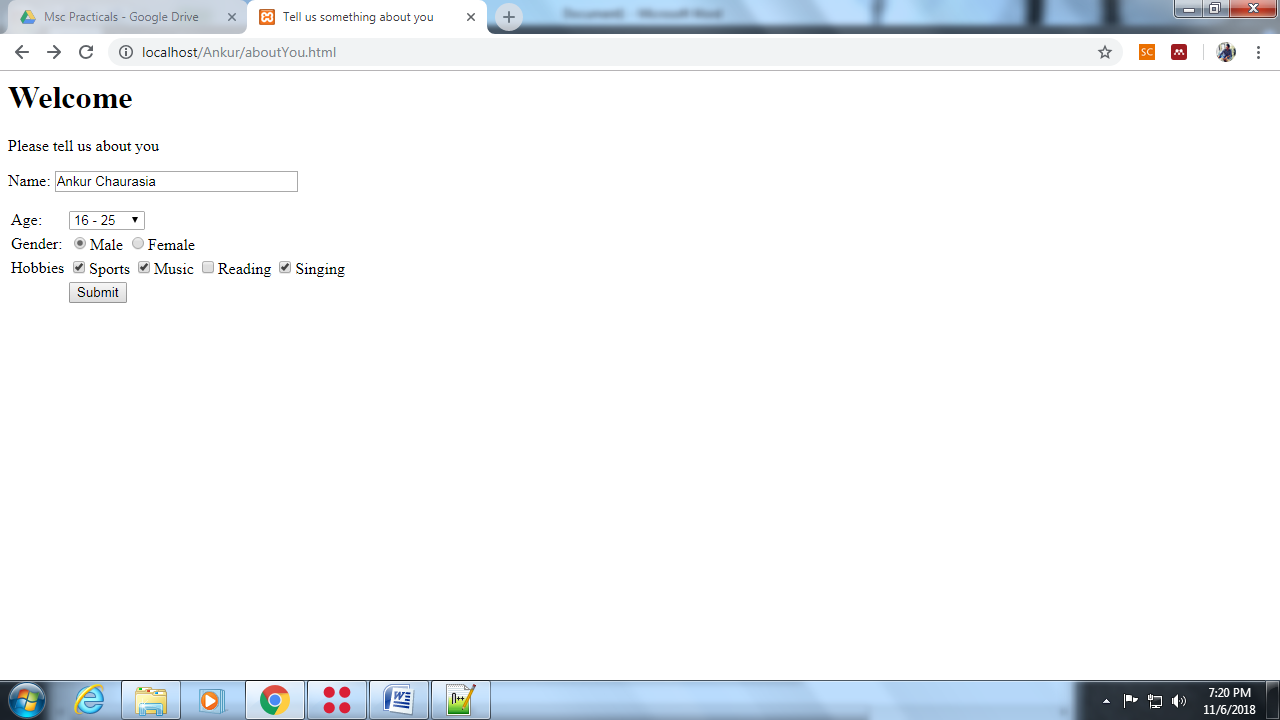
</table>

</form>

</body>

</html>

**OUTPUT**

****

**CGI CODE**

#!"C:\xampp\perl\bin\perl.exe"

use CGI(':standard');

print header;

my $name = param("name");

my $age = param("age");

my $gender = param("gender");

my @hobby = param("hobby");

my $str;

if (@hobby) {

$str = join( ', ', @hobby );

}

else {

$str = "none";

}

print start\_html( -title => "About You" ),

h1("Welcome $name"), p("Here are your details:"),

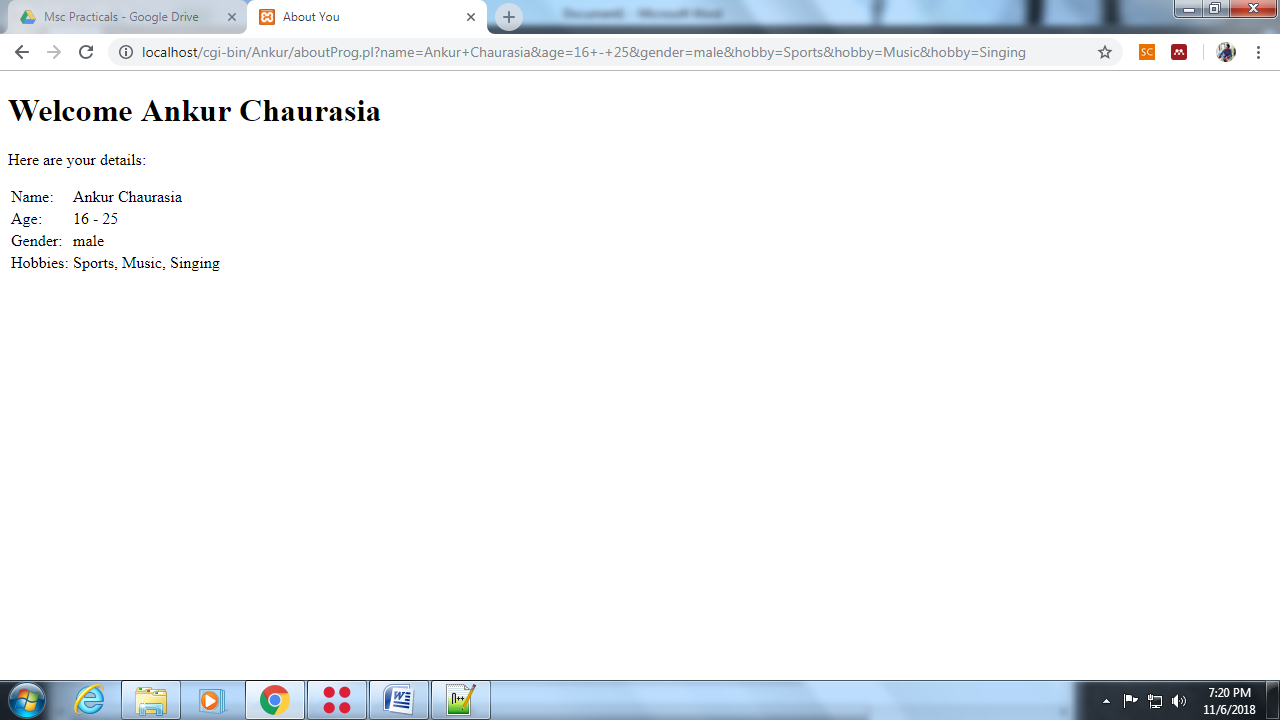
table(

Tr( td("Name:"), td($name) ),

Tr( td("Age:"), td($age) ),

Tr( td("Gender:"), td($gender) ),

Tr( td("Hobbies:"), td($str) ) ), end\_html;

**OUTPUT**

**PROGRAM 7**

**AIM**

**To write a CGI script to convert a DNA to a RNA sequence and count its nucleotide bases.**

**SOURCE CODE**

**HTML CODE**

<html>

<head>

<title>DNA to RNA</title>

</head>

<body>

<h1>DNA to RNA Convertor</h1>

<p>Enter the DNA sequence in the box given below:</p>

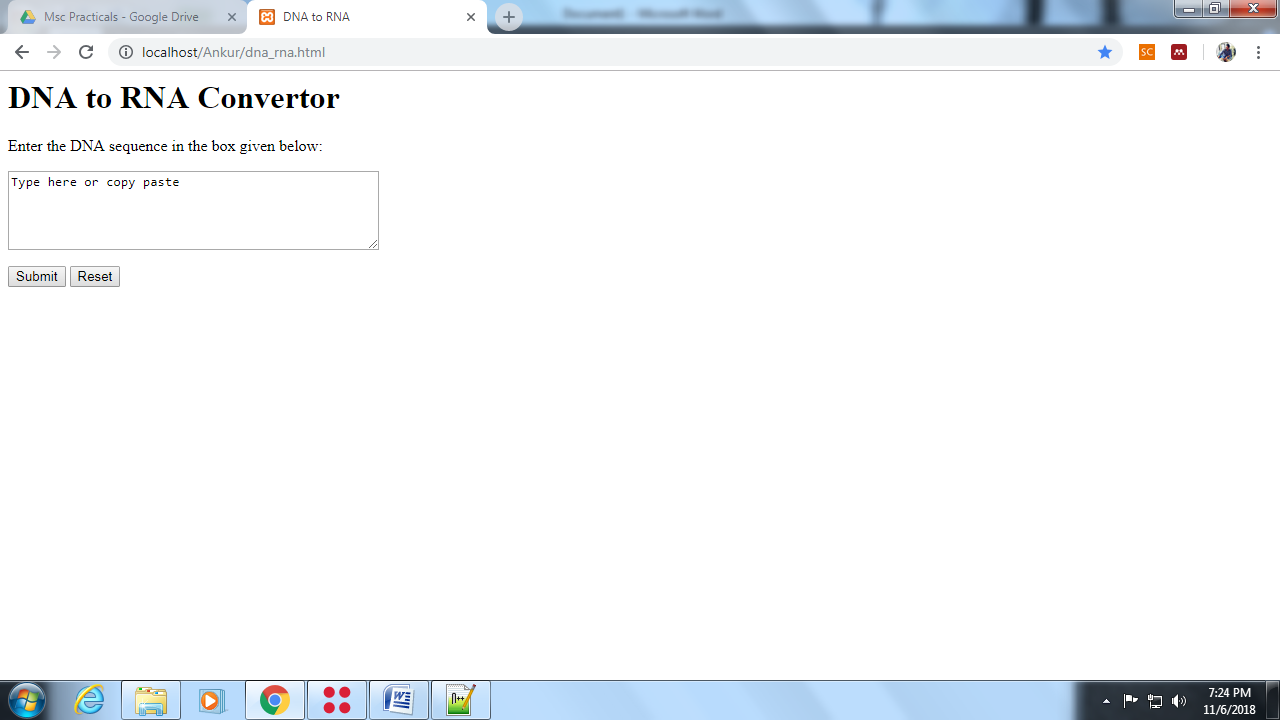
<form action="/cgi-bin/Ankur/dnaToRna.pl" method="post">

<textarea rows=5 cols=50 name="input">Type here or copy paste

</textarea>

<p><input type="submit" value="Submit"> <input type="reset" value="Reset"></p>

</form>

**OUTPUT**

**CGI CODE**

#!"C:\xampp\perl\bin\perl.exe"

use CGI(':standard');

my $seq = param("input");

$seq = uc($seq);

$dna = $seq;

#convert to RNA

$seq =~ tr/T/U/;

#counting bases

$count = ( $seq =~ tr/AUGC// );

print header;

print "<html>";

print "<head><title>DNA to RNA</title></head>";

print "<center><body bgcolor='silver'>";

print "<h1>DNA to RNA Convertor</h1><br><hr><br>";

print "<p>Here are the results:</p>";

print "<p>\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*</p>";

print "<p>Your protein has $count nucleotides</p>";

print "<p>\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*</p>";

print "<p>Your entered sequence is:</p>";

print

"<div style='width:500px;word-break:break-all;border:2px solid red'>$dna</div>";

print "<p>\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*</p>";

print "<p>Your RNA sequence is:</p>";

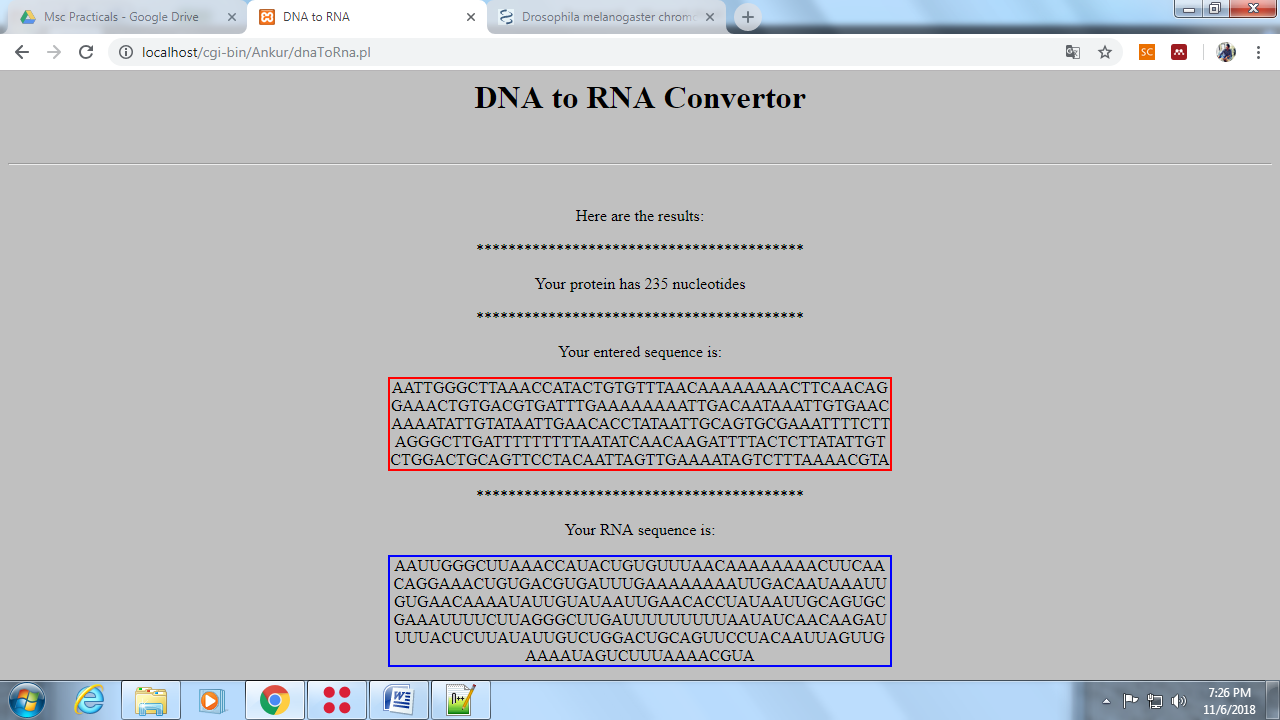
print

"<div style='width:500px;word-break:break-all;border:2px solid blue'>$seq</div>";

print "</body></center>";

print "</html>";

**OUTPUT**

****

**PROGRAM 8**

**AIM**

**To write a program to retrieve atom, heteroatom, helix and sheets parts from a PDB file.**

**SOURCE CODE**

#START

#!/usr/bin/perl

$file = "5ivl.txt";

open( IN, "<", $file ) || die "Can't open $file because $!.";

$file1 = "Atom.txt";

$file2 = "Helix.txt";

$file3 = "Sheet.txt";

$file0 = "Residues.txt";

$file4 = "Hetatom.txt";

open( ATOM, ">", $file1 ) or die "Can't open $file1 because $!.";

open( RES, ">", $file0 ) or die "Can't open $file0 because $!.";

open( HEL, ">", $file2 ) or die "Can't open $file2 because $!.";

open( SHE, ">", $file3 ) or die "Can't open $file3 because $!.";

open( HET, ">", $file4 ) or die "Can't open $file1 because $!.";

%Amino = (

"ASP" => "D",

"THR" => "T",

"SER" => "S",

"GLU" => "E",

"PRO" => "P",

"GLY" => "G",

"ALA" => "A",

"CYS" => "C",

"VAL" => "V",

"MET" => "M",

"ILE" => "I",

"LEU" => "L",

"TYR" => "Y",

"PHE" => "F",

"HIS" => "H",

"LYS" => "K",

"ARG" => "R",

"TRP" => "W",

"GLN" => "Q",

"ASN" => "N"

);

@data = <IN>;

$len = @data;

for ( $i = 0 ; $i < $len ; $i++ ) {

chomp( $data[$i] );

if ( $data[$i] =~ /^ATOM/ || $data[$i] =~ /^HETATOM/ ) {

print ATOM "$data[$i]\n";

}

elsif ( $data[$i] =~ /^HETATOM/ ) {

print HET "$data[$i]\n";

}

elsif ( $data[$i] =~ /^SEQRES/ ) {

@residues = split( " ", $data[$i] );

$length = @residues;

for ( $j = 4 ; $j < $length ; $j++ ) {

$char = $residues[$j];

print RES "$Amino{$char} ";

}

}

elsif ( $data[$i] =~ /^HELIX/ ) { print HEL "$data[$i]\n"; }

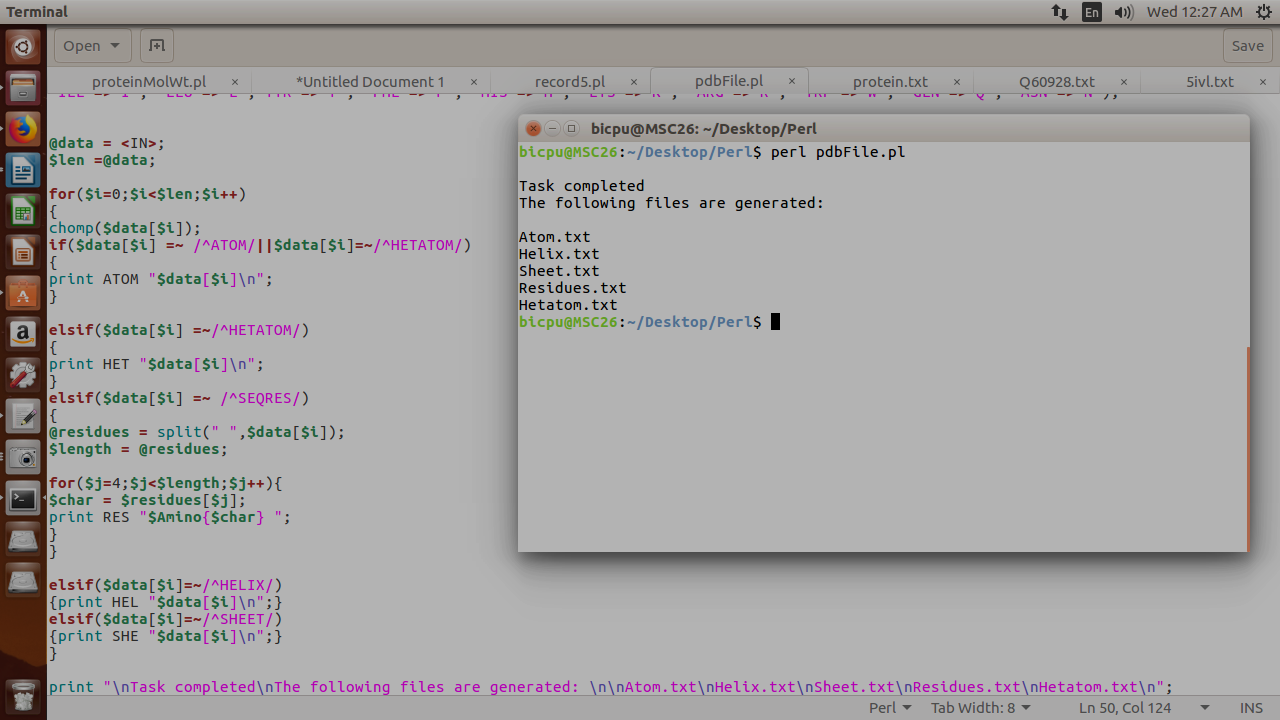
elsif ( $data[$i] =~ /^SHEET/ ) { print SHE "$data[$i]\n"; }

}

print

"\nTask completed\nThe following files are generated: \n\nAtom.txt\nHelix.txt\nSheet.txt\nResidues.txt\nHetatom.txt";

#END

**OUTPUT**

**PROGRAM 9**

**AIM**

**To retrieve the sequence by accession number using BioPerl.**

**SOURCE CODE**

#START

use Bio::Perl;

use Bio::SeqIO;

use Bio::DB::GenBank;

$genBank = new Bio::DB::GenBank;

my $seq = $genBank->get\_Seq\_by\_acc('KP191585.1');

my $dna = $seq->seq();

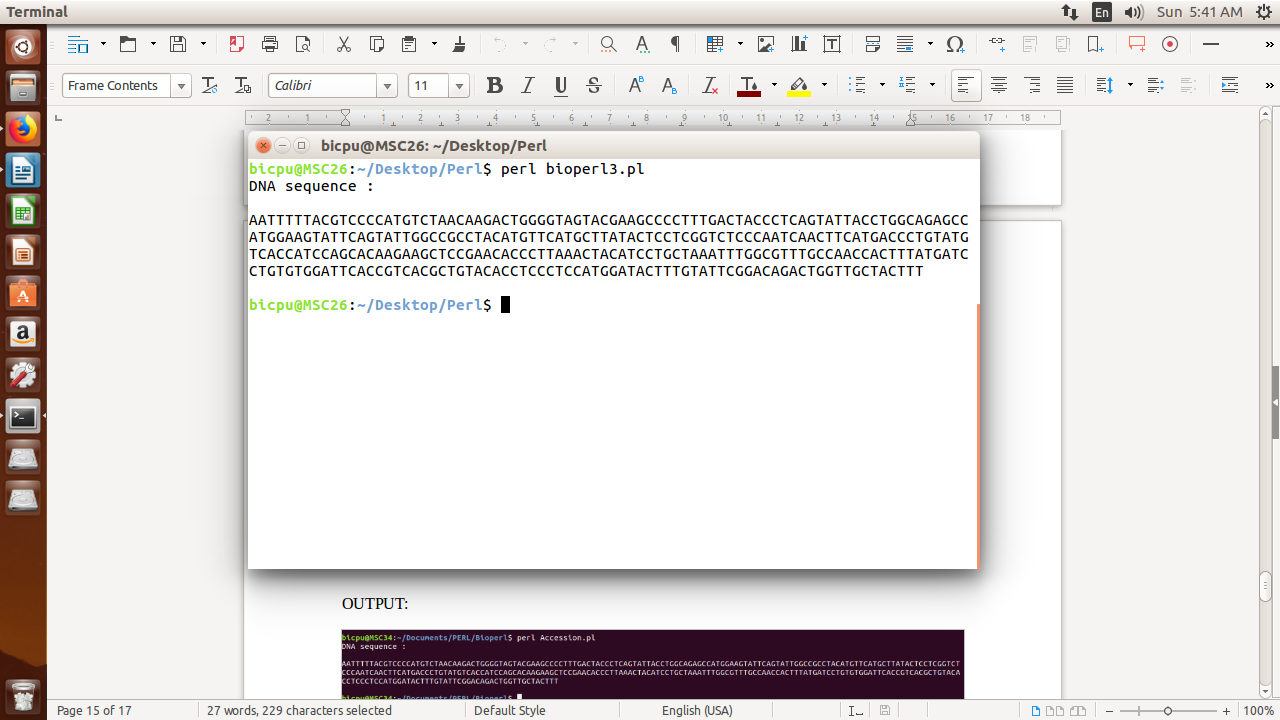
$dna=uc($dna);

chomp($dna);

print "DNA sequence : \n\n$dna\n\n";

#END

**OUTPUT**

****

**PROGRAM 10**

**AIM**

**To convert file format from GenBank to FASTA using Bioperl.**

**SOURCE CODE**

#START

use Bio::SeqIO;

$inFile = "dna2.fasta";

$in = Bio::SeqIO->newFh('-file' => "$inFile" ,

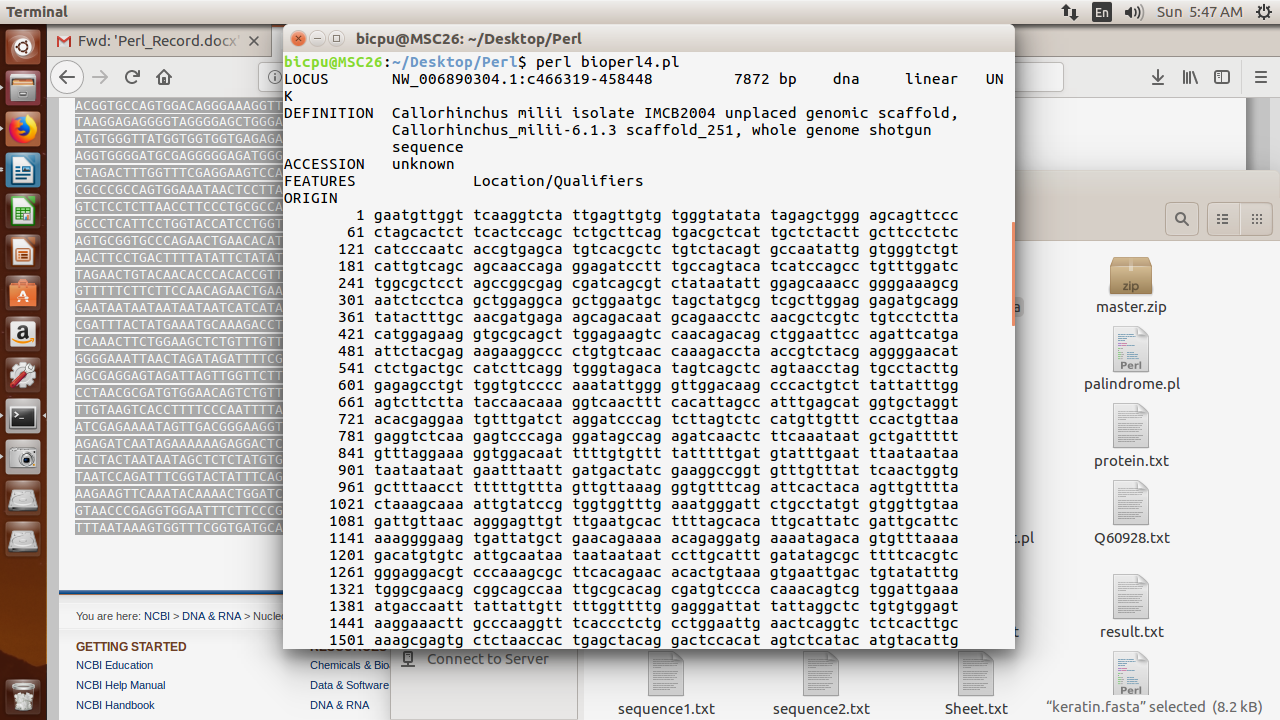
'-format' => 'Fasta');

$out = Bio::SeqIO->newFh('-format' => 'Genbank');

print $out $\_ while <$in>;

#END

**OUTPUT**

****