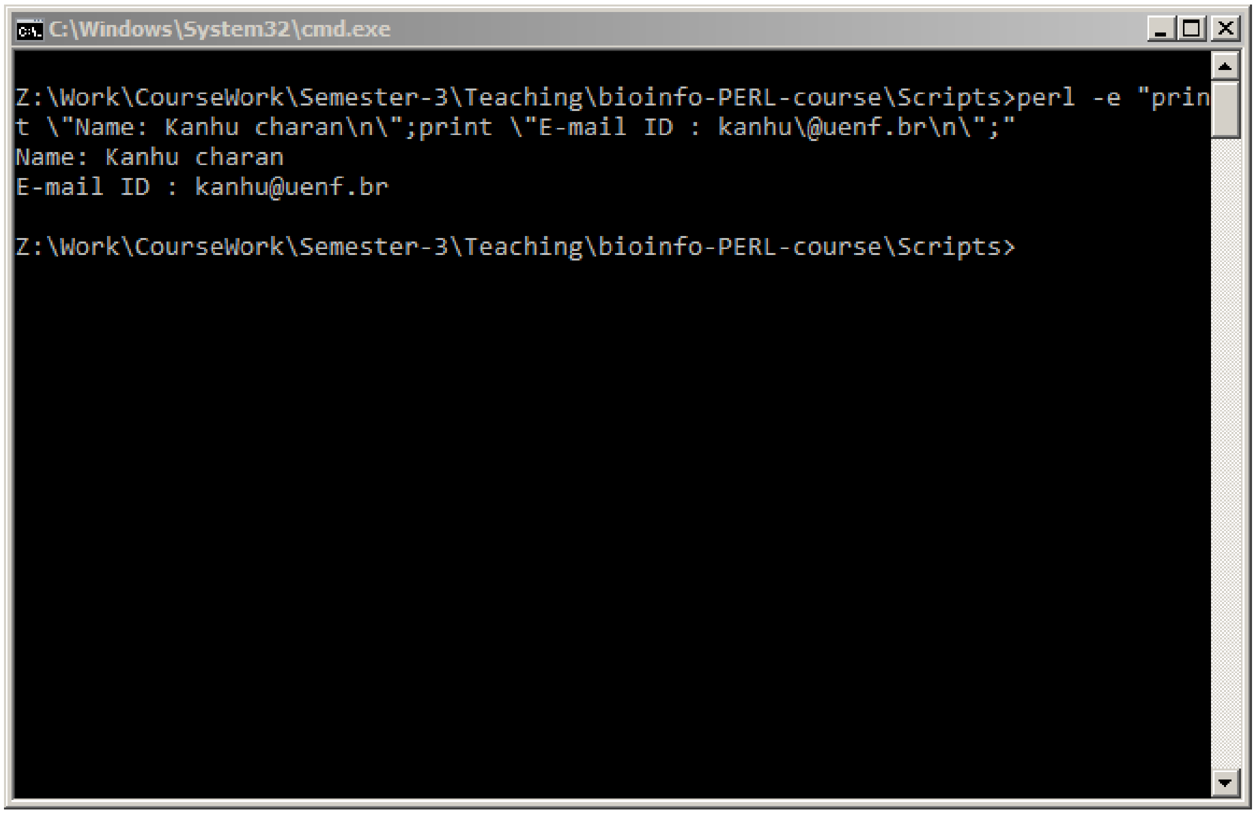
# Solution to 1.1.Getting_stared_with_perl

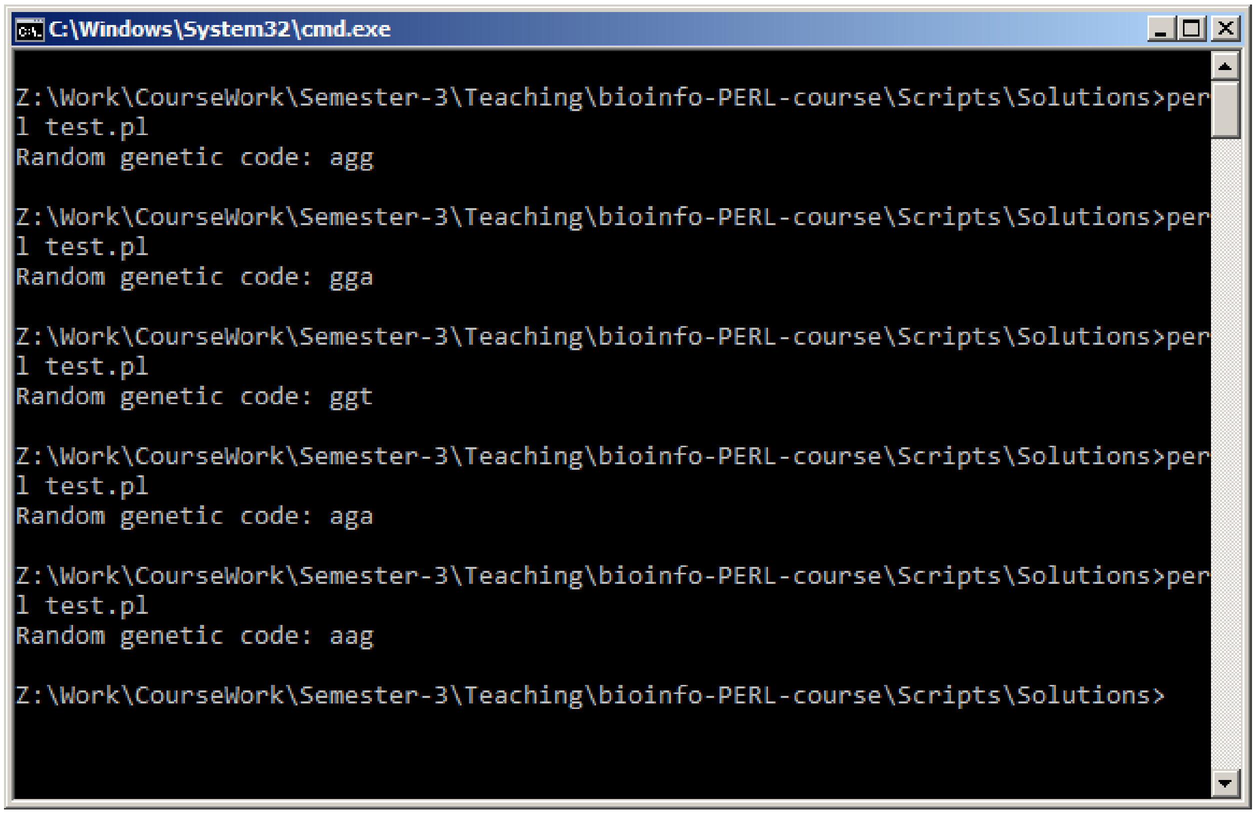
```
## Check and report version of  PERL on your computer.
#Ans : open command prompt. Type perl -v


## Write a Perl program that prints your name and e-mail in the
# following format
     #Name: Student_name
     #E-mail ID : student@uenf.br
#Ans:  write the follwing code in a text file, save the file and run
 use warnings;
 print "Name: Kanhu charan";
 print "\n";
  # use escape character \ before @
 print "E-mail ID : kanhu\@uenf.br";
 print "\n";


## Write the same program as an one liner script.

#Ans: On command prompt type the follwoing; and press Enter
# on Linux (use single quotes)
perl -e 'print "Name: Kanhu charan\n";print "E-mail ID : kanhu\@uenf.br\n";'

# on Windows (use double quotes and escape character)
perl -e "print \"Name: Kanhu charan\n\";print \"E-mail ID :
kanhu\@uenf.br\n\";"
```



```
Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts>perl -e "prin
t \"Name: Kanhu charan\n\";print \"E-mail ID : kanhu\@uenf.br\n\";"
Name: Kanhu charan
E-mail ID : kanhu@uenf.br

Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts>
```

# Solutions to 5.Operations_on_variables

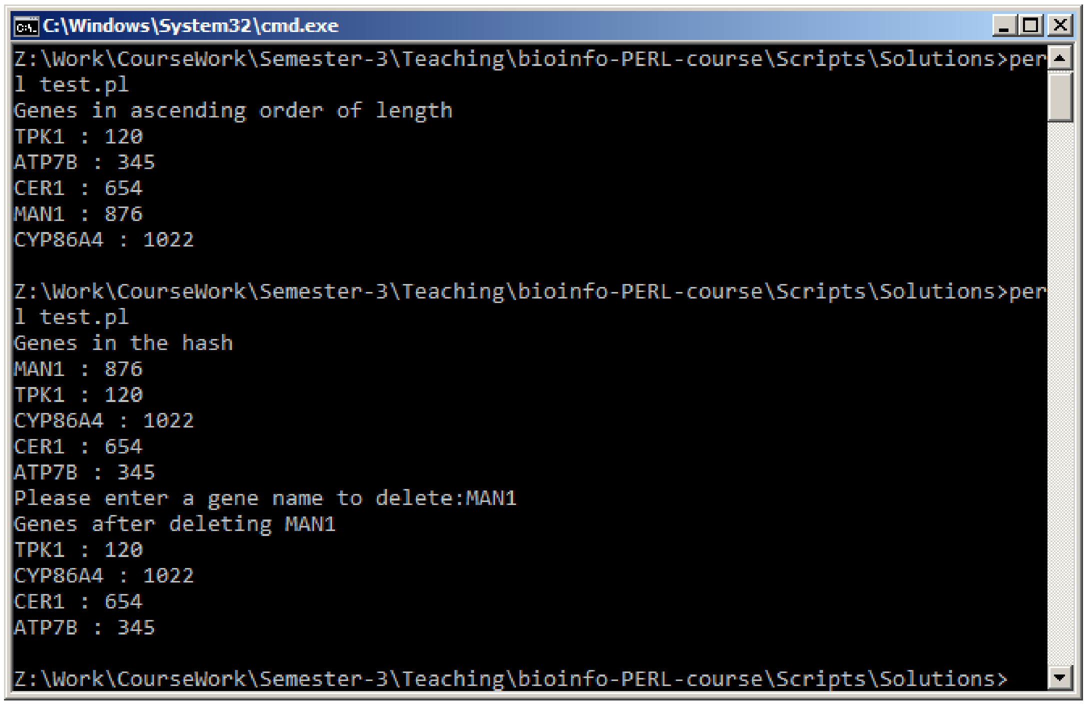
```
## Ask user to input a DNA sequence and print it in upper case letters.
#Ans:

use warnings;
print "Enter a DNA sequence: ";
$dna = <STDIN>;
$dna_uc = uc $dna;
print "DNA in upper case : $dna_uc\n";

## Ask user to input a DNA sequence and print its first 3 nucleotides
#   and last 3 nucleotides.

use warnings;
print "Enter a DNA sequence: ";
$dna = uc <STDIN>;
chomp($dna);
$start_3 = substr($dna, 0,3);# position , length
$end_3 = substr($dna, -3); ## negative position
print "DNA seq: $dna\nFirst three: $start_3\nLast three base: $end_3\n";




## You were provided with an array @DNA=('a','t','g','c');
# Generate a random genetic code by combaining any 3 nucleotide bases.
# Write a program to generate and print a random amino acid
# translated from the genetic code.
# (Hint: assume have stored genetic codes in an hash %genetic_code)

#Ans:
use warnings;
@DNA = ('a','t','g','c');
$three_base = $DNA[int rand(3)].$DNA[int rand(3)].$DNA[int rand(3)];
print "Random genetic code: $three_base\n";
# you may comment the follwoing code, as it will warn as error
print "Amino acid coded by $three_base: $genetic_code{$three_base}\n";
```



```
Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>per
l test.pl
Random genetic code: agg

Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>per
l test.pl
Random genetic code: gga

Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>per
l test.pl
Random genetic code: ggt

Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>per
l test.pl
Random genetic code: aga

Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>per
l test.pl
Random genetic code: aag

Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>
```

```
## Create an hash of 5 genes.
#  Gene name as key and their lengths as values.
#  Print the list of genes with increasing order of gene length.

use warnings;

%genes = (
 ATP7B => 345,
 CYP86A4 => 1022,
 CER1 => 654,
 TPK1 => 120,
 MAN1 => 876,
);
# numerically sorting by values
@genes_or = sort {$genes{$a}<=> $genes{$b}} keys %genes;
print "Genes in ascending order of length\n";
print "$genes_or[0] : $genes{$genes_or[0]}\n";
print "$genes_or[1] : $genes{$genes_or[1]}\n";
print "$genes_or[2] : $genes{$genes_or[2]}\n";
print "$genes_or[3] : $genes{$genes_or[3]}\n";
print "$genes_or[4] : $genes{$genes_or[4]}\n";
```

```perl
## Use the same gene hash to display the gene list to the user and ask to
type a gene name.
# Write a program  to delete user defined gene from the hash.
# Print the updated gene list.
use warnings;

%genes = (
 ATP7B => 345,
 CYP86A4 => 1022,
 CER1 => 654,
 TPK1 => 120,
 MAN1 => 876,
);
@genes_or = keys %genes;
print "Genes in the hash \n";
print "$genes_or[0] : $genes{$genes_or[0]}\n";
print "$genes_or[1] : $genes{$genes_or[1]}\n";
print "$genes_or[2] : $genes{$genes_or[2]}\n";
print "$genes_or[3] : $genes{$genes_or[3]}\n";
print "$genes_or[4] : $genes{$genes_or[4]}\n";

print "Please enter a gene name to delete:";
chomp($g = <STDIN>);
delete $genes{$g};
# get the keys
@genes_or = keys %genes;
print "Genes after deleting $g\n";
print "$genes_or[0] : $genes{$genes_or[0]}\n";
print "$genes_or[1] : $genes{$genes_or[1]}\n";
print "$genes_or[2] : $genes{$genes_or[2]}\n";
print "$genes_or[3] : $genes{$genes_or[3]}\n";
```



```
Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>per
l test.pl
Genes in ascending order of length
TPK1 : 120
ATP7B : 345
CER1 : 654
MAN1 : 876
CYP86A4 : 1022

Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>per
l test.pl
Genes in the hash
MAN1 : 876
TPK1 : 120
CYP86A4 : 1022
CER1 : 654
ATP7B : 345
Please enter a gene name to delete:MAN1
Genes after deleting MAN1
TPK1 : 120
CYP86A4 : 1022
CER1 : 654
ATP7B : 345

Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>
```

# Solutions to 7.Perl_file_handling

```
## Write a Perl program that asks user to enter a file name and a target file
name. Copy the content of the first file to the second one.

use warnings;

print "Please enter a file name to copy:";
$file = <STDIN>;
chomp($file);

print "Please enter a target file name:";
$file_o = <STDIN>;
chomp($file_o);

open IN, "< $file" or die "$!, $file";
open COPY, "> $file_o" or die "$!, $file_o";

while ($line=<IN>){
 print COPY $line ;
}
# close filehandles
close IN; close COPY;
print STDERR "$file copied to $file_o successfully\n";
```

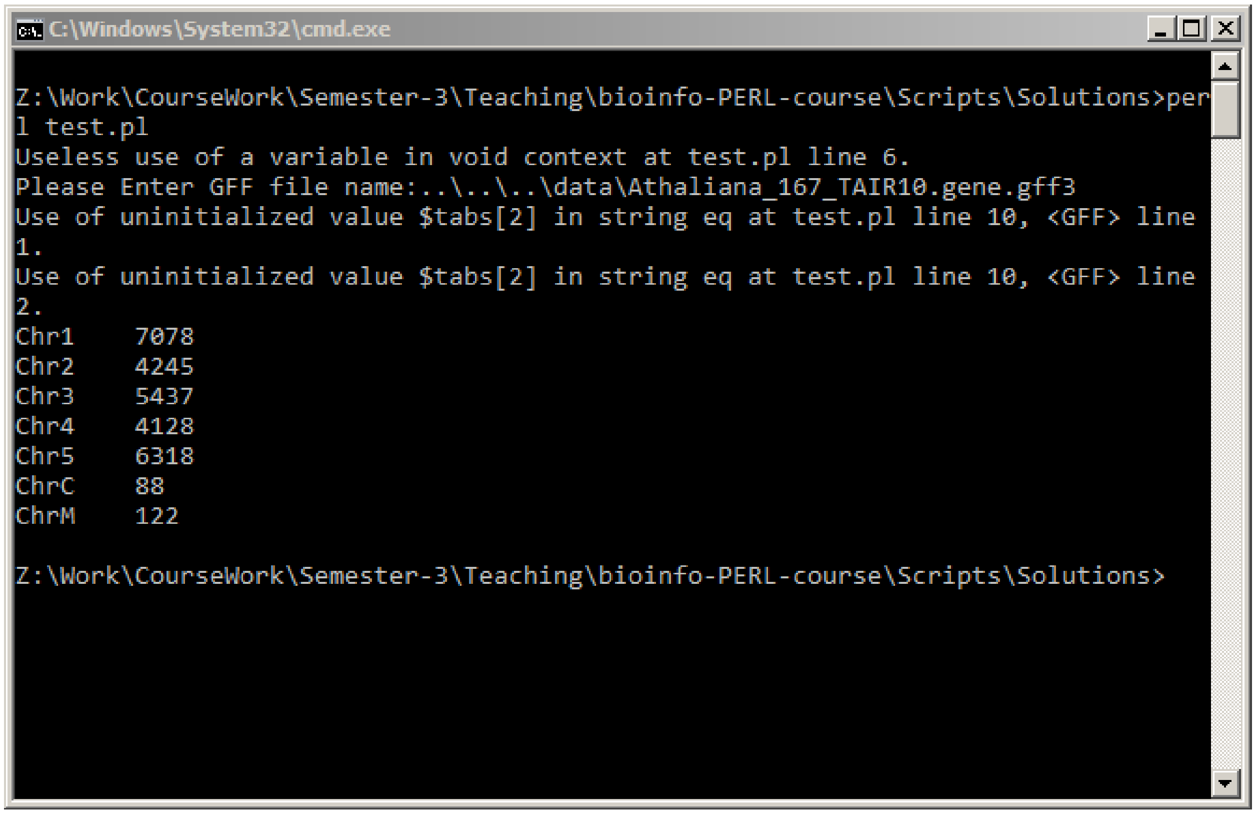
```
#Write a Perl script to parse a BlastP tabular output file and save those
# hits with >45% identity.

use warnings;

$blast_out = "blastp_output.txt";

open IN, "< $blast_out" or die "$!, $blast_out";

while ($line=<IN>){
# split each line into an array
 @record = split /\t/,$line;
 if($record[2]>=35)  ## check the third column or the 2-index of the array
>35
 {
      print "$line";
 }
}
# close filehandles
close IN;
```

```perl
## Write a Perl script that asks user to enter a Directory name. Print only
the files present in the directory (excluding the directories).

use warnings;

print "Enter a directory name: ";
$dir = <STDIN>;
chomp($dir);
opendir DIR, "$dir" or die "Error: $!, $dir\n";
      @ent = readdir(DIR);
closedir DIR;
print "Files present in $dir are : \n";
foreach $f(@ent){
 print "$dir\\$f\n" if (-f "$dir\\$f");
}
```

```perl
## Open a FASTA file and copy its content to another file.
#Ans: Similar to copy files example
```

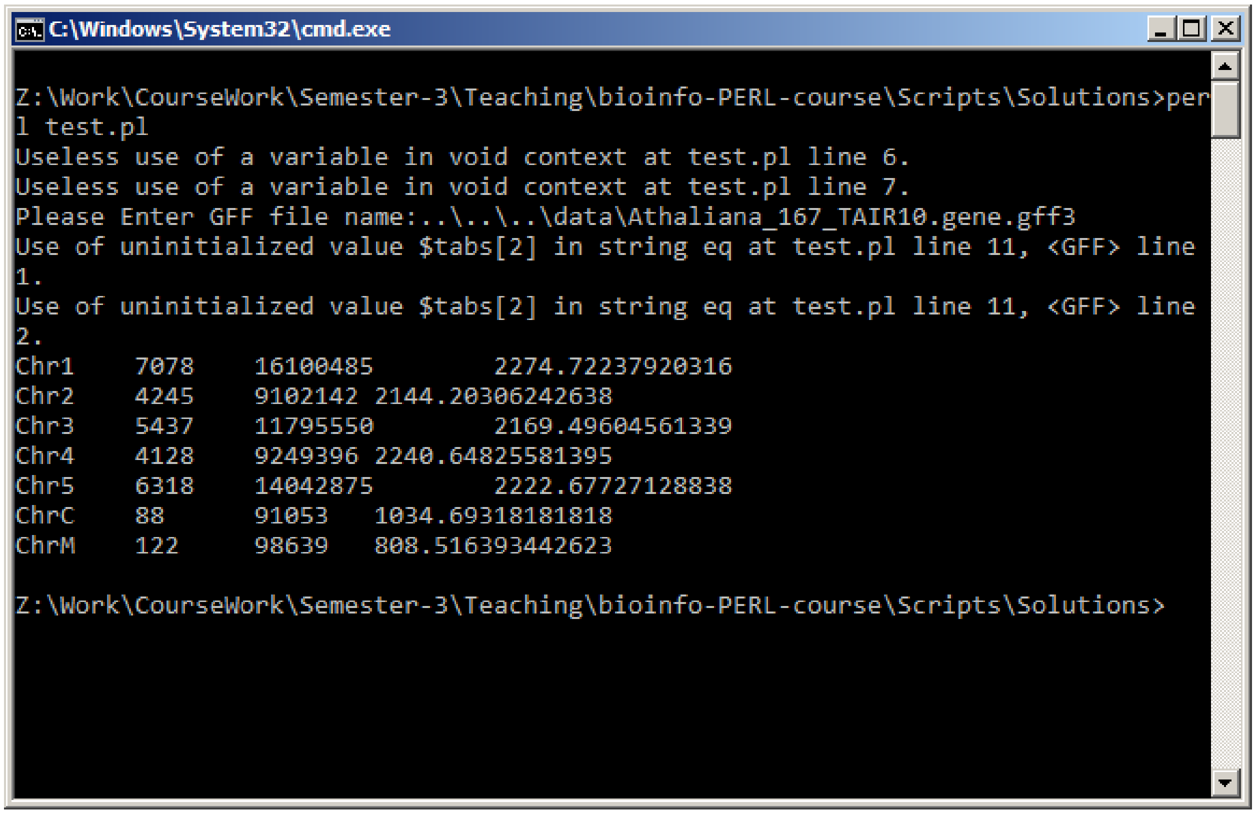
```perl
## Ask user to input to two file names and add '__END_OF_FILE__' to both of
these files
#Ans:

use warnings;

print "Please enter first file name:";
$file1 = <STDIN>;
chomp($file1);

print "Please enter second file name:";
$file2 = <STDIN>;
chomp($file2);

open F1, ">> $file1" or die "$!, $file1";
open F2, ">> $file2" or die "$!, $file2";
print F1 "__END_OF_FILE__\n";
print F2 "__END_OF_FILE__\n";
# close filehandles
close F1; close F2;
```

```perl
## Open a GFF file and report total number of Genes present.
#Ans:
use warnings;

print "Please Enter GFF file name:";
$gff = <STDIN>;
chomp($gff);
$gene_count = 0;
open GFF, "< $gff" or die "$!, $gff";
while($l=<GFF>){
 @tabs = split /\t/,$l;
 if($tabs[2] eq 'gene'){
 $gene_count++;
 }
}
# close filehandles
close GFF;
print "Total number of genes in $gff : $gene_count\n";

# Answer: for Athaliana.gff3: 27416



## Open a GFF file and report total number of Genes present per chromosome.
#Ans:
use warnings;

print "Please Enter GFF file name:";
$gff = <STDIN>;
chomp($gff);
%gene_count; # create a hash to count for each Chromosome
open GFF, "< $gff" or die "$!, $gff";
while($l=<GFF>){
 @tabs = split /\t/,$l;
 if($tabs[2] eq 'gene'){
 $gene_count{$tabs[0]}++; ## First column contains Chromsome name
 }
}
# close filehandles
close GFF;
foreach $chr(sort keys %gene_count)
{
      print "$chr\t$gene_count{$chr}\n";
}
```



```
C:\Windows\System32\cmd.exe

Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>per
l test.pl
Useless use of a variable in void context at test.pl line 6.
Please Enter GFF file name:..\..\..\data\Athaliana_167_TAIR10.gene.gff3
Use of uninitialized value $tabs[2] in string eq at test.pl line 10, <GFF> line
1.
Use of uninitialized value $tabs[2] in string eq at test.pl line 10, <GFF> line
2.
Chr1    7078
Chr2    4245
Chr3    5437
Chr4    4128
Chr5    6318
ChrC    88
ChrM    122

Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>
```

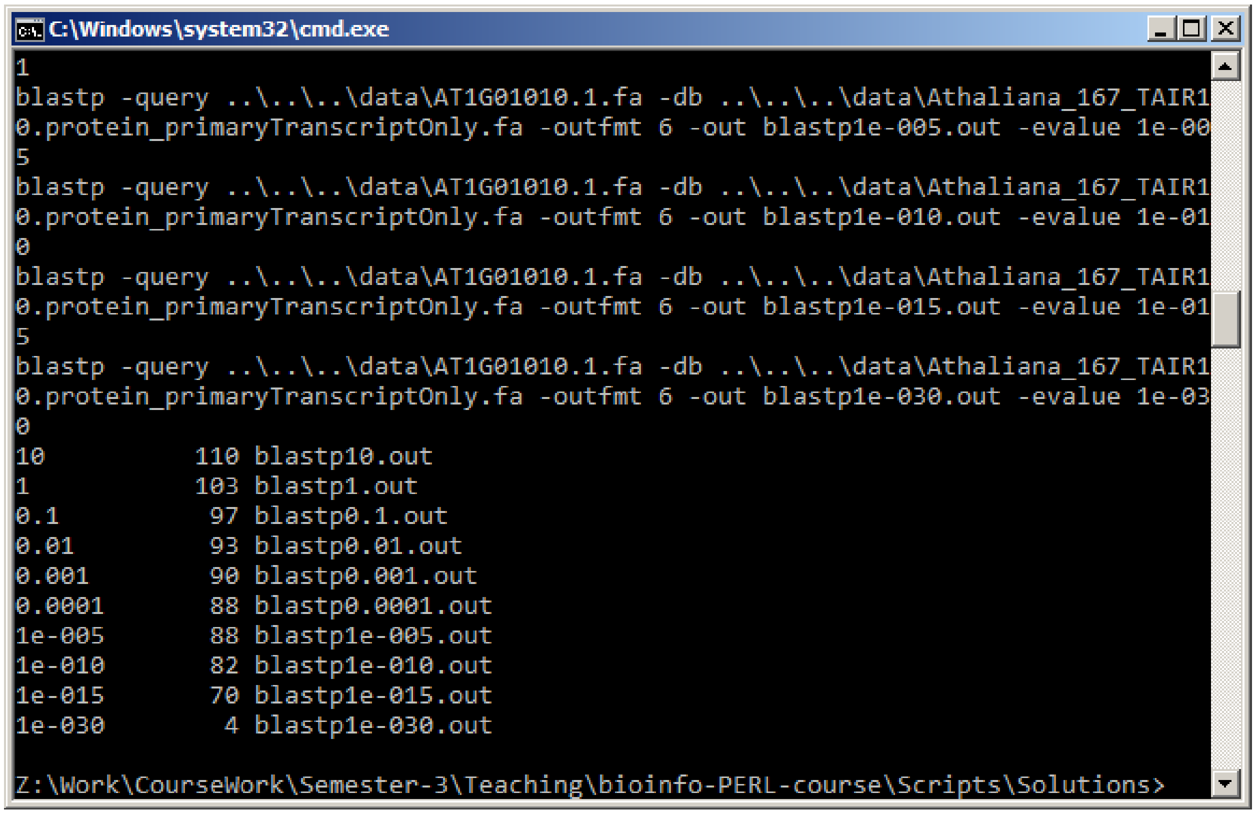
```
## Open a GFF file and report average legth of genes per chromosome.
use warnings;

print "Please Enter GFF file name:";
$gff = <STDIN>;
chomp($gff);
%gene_count; # create a hash to count for each Chromosome
%gene_length; # create a hash to total gene length for each Chromosome
open GFF, "< $gff" or die "$!, $gff";
while($l=<GFF>){
 @tabs = split /\t/,$l;
 if($tabs[2] eq 'gene'){
 $gene_count{$tabs[0]}++; ## First column contains Chromsome name
 $gene_length{$tabs[0]}+=$tabs[4]-$tabs[3]+1; #4,5 column contains gene start
and end locations
 }
}
close GFF;

foreach $chr(sort keys %gene_count)
{
      print "$chr\t$gene_count{$chr}\t$gene_length{$chr}\t";
      $avg = $gene_length{$chr}/$gene_count{$chr};
      print "$avg\n";
}
```



```
Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>per
l test.pl
Useless use of a variable in void context at test.pl line 6.
Useless use of a variable in void context at test.pl line 7.
Please Enter GFF file name:..\..\..\data\Athaliana_167_TAIR10.gene.gff3
Use of uninitialized value $tabs[2] in string eq at test.pl line 11, <GFF> line
1.
Use of uninitialized value $tabs[2] in string eq at test.pl line 11, <GFF> line
2.
Chr1    7078    16100485        2274.72237920316
Chr2    4245    9102142 2144.20306242638
Chr3    5437    11795550        2169.49604561339
Chr4    4128    9249396 2240.64825581395
Chr5    6318    14042875        2222.67727128838
ChrC    88      91053   1034.69318181818
ChrM    122     98639   808.516393442623

Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>
```

Solutions to 10.Run_external_programs

```
## Write a script to perform blastP search at 10 different e-value
#  cutoff. Report total number of hits predicted in each search.
#Ans:

use warnings;
use strict;

my @e_values = (10,1,0.1,0.01,0.001,0.0001,0.00001,1e-10, 1e-15, 1e-30);
my $query = $ARGV[0];
my $db = $ARGV[1];
my %h; # to store no of hits for each e-val
foreach my $e(@e_values){
 my $cmd = "blastp -query $query -db $db -outfmt 6 -out blastp$e\.out -evalue
$e";
 print STDERR "$cmd\n";
 system($cmd);
 my $hit = `wc -l blastp$e\.out`;
 chomp($hit);
 $h{$e} = $hit;
 unlink ("$query\_blastp.out");
}

foreach my $e(@e_values){
 print "$e\t$h{$e}\n";
}
```



```
1
blastp -query ..\..\..\data\AT1G01010.1.fa -db ..\..\..\data\Athaliana_167_TAIR1
0.protein_primaryTranscriptOnly.fa -outfmt 6 -out blastp1e-005.out -evalue 1e-00
5
blastp -query ..\..\..\data\AT1G01010.1.fa -db ..\..\..\data\Athaliana_167_TAIR1
0.protein_primaryTranscriptOnly.fa -outfmt 6 -out blastp1e-010.out -evalue 1e-01
0
blastp -query ..\..\..\data\AT1G01010.1.fa -db ..\..\..\data\Athaliana_167_TAIR1
0.protein_primaryTranscriptOnly.fa -outfmt 6 -out blastp1e-015.out -evalue 1e-01
5
blastp -query ..\..\..\data\AT1G01010.1.fa -db ..\..\..\data\Athaliana_167_TAIR1
0.protein_primaryTranscriptOnly.fa -outfmt 6 -out blastp1e-030.out -evalue 1e-03
0
10          110 blastp10.out
1           103 blastp1.out
0.1          97 blastp0.1.out
0.01         93 blastp0.01.out
0.001        90 blastp0.001.out
0.0001       88 blastp0.0001.out
1e-005       88 blastp1e-005.out
1e-010       82 blastp1e-010.out
1e-015       70 blastp1e-015.out
1e-030        4 blastp1e-030.out

Z:\Work\CourseWork\Semester-3\Teaching\bioinfo-PERL-course\Scripts\Solutions>
```

```
## Write a script to automate  BLASTp search, using each of the
#  fasta files in the user defined directory against A.thaliana proteome.
#Ans:
use warnings;
use strict;


my $query = $ARGV[0]; ## Enter directory containing fasta files
my $db = $ARGV[1];    # database file path

opendir DIR, "$query" or die "Error: $! $query";
my @files = readdir(DIR);
closedir DIR;

foreach my $f(@files){
 if(-f "$query/$f" and $f=~m/\.faa$/){ # check is it a *.faa file
      my $cmd = "blastp -query $query/$f -db $db -outfmt 6 -out
blastp$f\.out";
      print "$cmd\n";
      system("$cmd");
      }
}
```