Bioinformatics

Prof. M. Michael Gromiha
Department of Biotechnology
Indian Institute of Technology, Madras

Lecture – 28a
"awk" Programming I

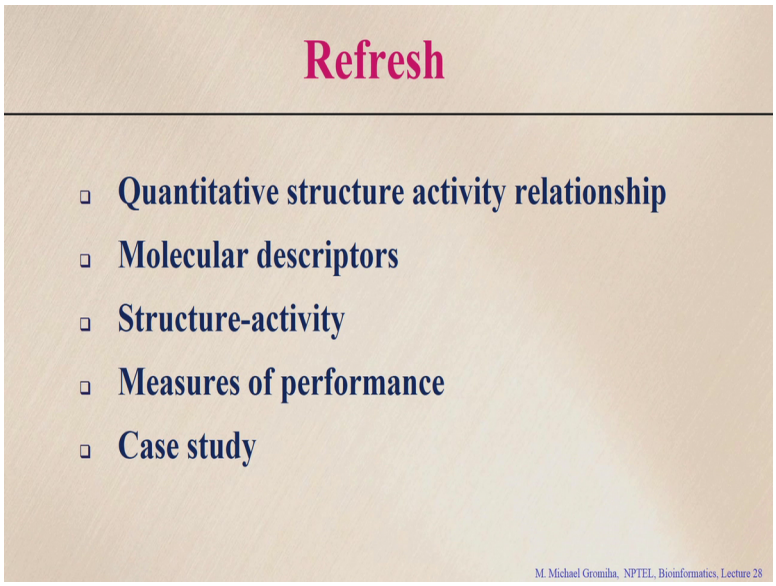
In this lecture, we will mainly discuss about awk programming, it is a script language; how to manipulate the data files or the output obtained from the unique programmings and so on.

So, in the past few classes we focused on different aspects of bioinformatics and the application of bioinformatics including the development of databases and algorithms, and how to deal with the different specific type of algorithms we developed for protein sequences or DNA sequences and so on.

Also we discussed about various properties or features derived from known amino acid sequences or from this 3D structures, and then we discussed about the applications of the different bioinformatics database and tools for predicting the 3D structure of a protein or folding rates or stability as well as protein interactions.

In the last class we discussed about the computer aided drug design and different aspects; such as docking or the pharmacophore modelling or we discussed about the virtual screening and the quantitative structure activity relationship.

(Refer Slide Time: 01:23)



In the quantitative structure activity relationship say QSAR. what are the various aspects we need to consider?

Student: (Refer Time: 01:34) biological activity.

Right mainly the main aspect is the biological activity can be described as a function of the molecular descriptors, like the structural parameters right. So, you can use various types of descriptors right for example, the 1d QSAR like molecular weight or hydrogen bond donor and so on. And they can use the connectivity as well as you can give the information regarding the surrounding residues in the environment.

Then we can relate these parameters with the activity and to measure the performance, whether this we can get the good confidence or not right by means of the correlation. So, depending upon the correlation as well as the cross validations right then we discussed few examples mainly on the Bcl-2 and the EGFR, mutants for the cancer.

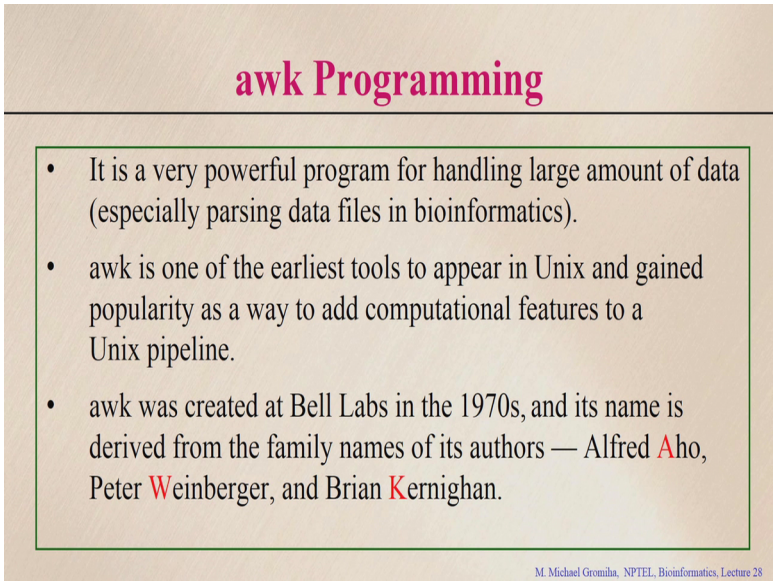
So, we try to see how we can identify new compound by fitting the QSAR models. In this class and the subsequent classes we discuss mainly about the algorithms as well as how we approach a problem, may it for a project or any type of analysis. So awk programming it is a scripting language, it is a kind of pattern matching program right for processing files because when you deal with the several databases or we write program

for analysing any structures or sequences, you will end up with getting a bulk amount of data.

So, doing the manipulating the data it takes time. So, in this case you can use this awk programming to get the desired output right in a very simple and easy way. See it is kind of pattern matching program for processing the files, especially if your databases or any output obtained from this programs. So, it is a data extraction reporting tool that uses a data driven scripting language right, which consists of set of actions to be taken against any data right for the purpose of producing output right.

You have any specific input data, you want to process the data with respect to any action you can give and; what do you want see the output you can format the output as per your desire. So, this is the main applications of this awk programming.

(Refer Slide Time: 03:53)



It is a simple programming, but it can it is very powerful to handle large amount of data especially parsing several data files in bioinformatics for example, we discussed about various databases right can you remember any of these databases?

Student: UniProt.

Protein sequence database: uniprot, protein structure database

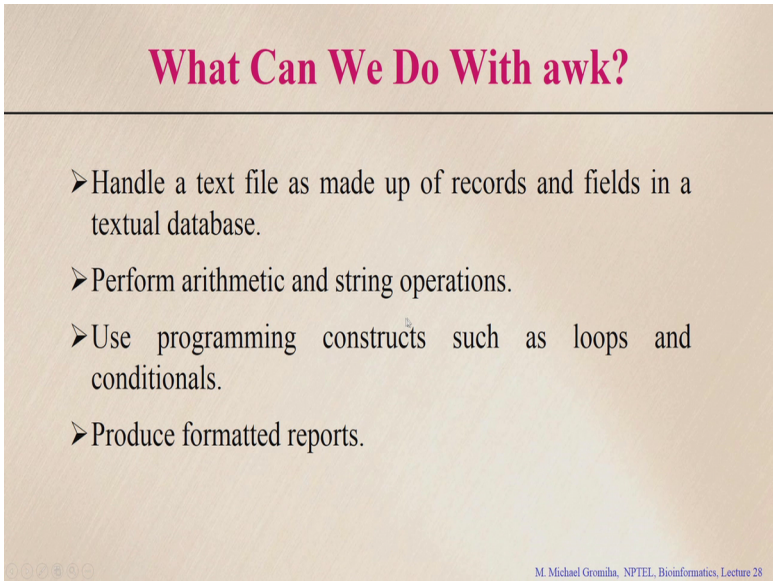
Student: PDB.

PDB right for example, if you are interested only on the coordinates in the PDB file right. It just if you are writing one line command right we can get only the coordinates in the PDB file without looking into these PDB file. If we do not have to open the PDB file just if you know the input name right in one line you can get all the information regarding the coordinates right. This is I will explain how to use this awk program to get the files.

So, it is very powerful program. So, it is one of the earliest tools to appear in Unix right, then we can get the popularity now to use the various other features and also you can use other combine other programs right to make a program using this awk. See if we talk about awk it was created in Bell Labs in 1970s right and the names came a w k that is from the family names of these authors. So, three persons that is Aho right and Weinberger and this Kernighan.

So, with the names they have put a w k. So, this how they made this initially they started the name awk right. So, what can we do with awk?

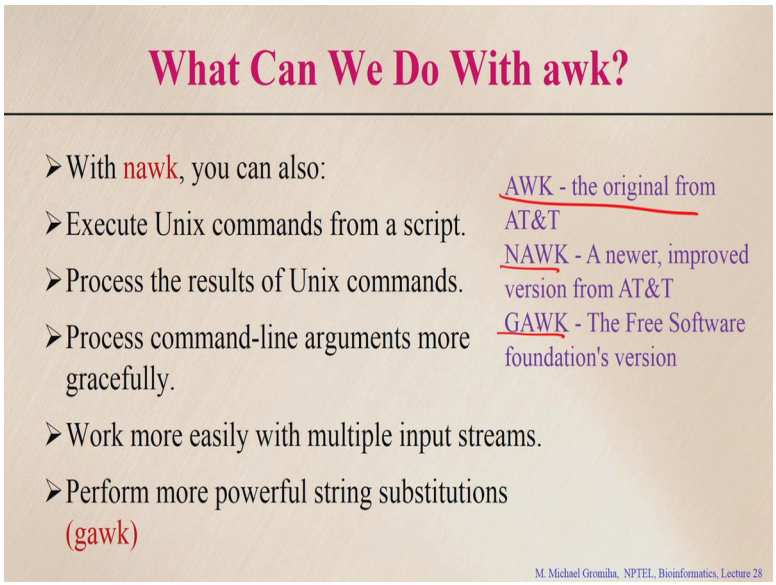
(Refer Slide Time: 05:19)



So, what are the various applications of awk? You can handle a text file right made up of any records and fields any text database, and you can perform the arithmetic and sorting of string operations you can also search with several strings, you can also do arithmetic operations right they have explained some examples how to do this operation.

And you can use a programming this by constructing loops and the conditions, you can also write a program using this awk right. So, let us also you can use to produce the formatted reports right in any specific format.

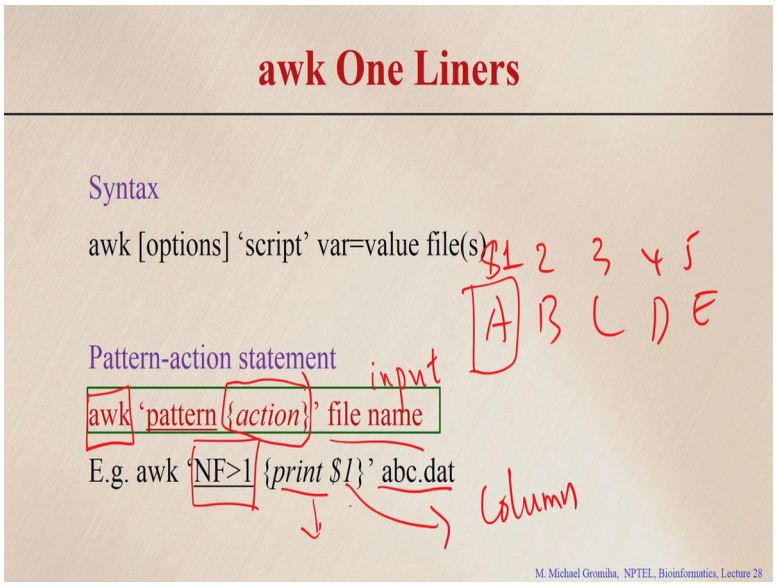
(Refer Slide Time: 05:52)



Then there are different types of awk right. The first original form, this is the AWK, the original form right then they developed new versions right one is the NAWK right this is newer and improved versions right. Using this version you can execute unix commands from a script, and you can process the results of unix commands, and also you can do the work more easily with multiple input steps.

Then if you take the GWAK this is another version of this awk this is a free software which is available from this foundations version. So, they are also perform very more powerful string substitutions than the previous versions like a simple awk or this nawk right. So, what is syntax?

(Refer Slide Time: 06:35)



How the program awk works right. So, it is kind of pattern action statement, this is syntax awks you can use some options and this is script you can use, variable equal to value of these files.

So, this is the pattern how to write this syntax for the awk? First you write "awk" this is the programming name "awk" and you need the input file name. So, right here this is your input file name we can put the input; and here the action; action means what you have to do right for example, here we have to do, the program tells we need to print. So, which one we need to print?

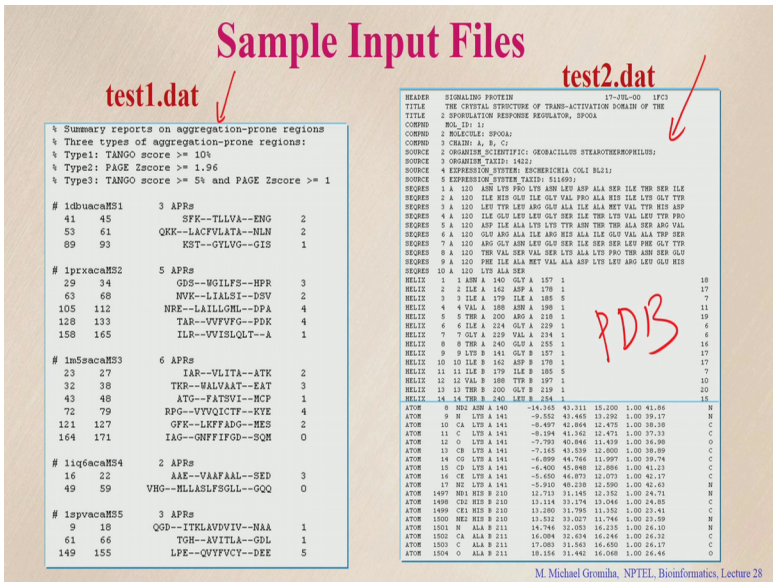
First column, the dollar one this, means a column right for example, if this is the A B C D E this is column 1, this is 2, 3, 4, 5. So, you use this dollar symbol to specify the column. So, this is the action means what we need to do that. So, here we have to print only the first column. So, first column means this is the first column you have to print this, and this is a pattern. Pattern means if it satisfies which conditions we need to print we need to do the action right we can use some conditions right for example, if there is no empty line if any line contains some characters or any line contains any numbers.

Any number contains, any line contains the numbers more than 50 right. You can give, find any pattern that. If it satisfies the pattern then do the action right here, you can see just print right this is a input file. So, here input file is abc dot dat. So, it considers the condition NF greater than 1 I will explain what is this right, this will check this condition

in this file abc dot dat and if it satisfies the condition then we will print the first column ok.

This is the syntax or this awk pattern action statement that is the awk. So, I show some of the input files.

(Refer Slide Time: 08:43)



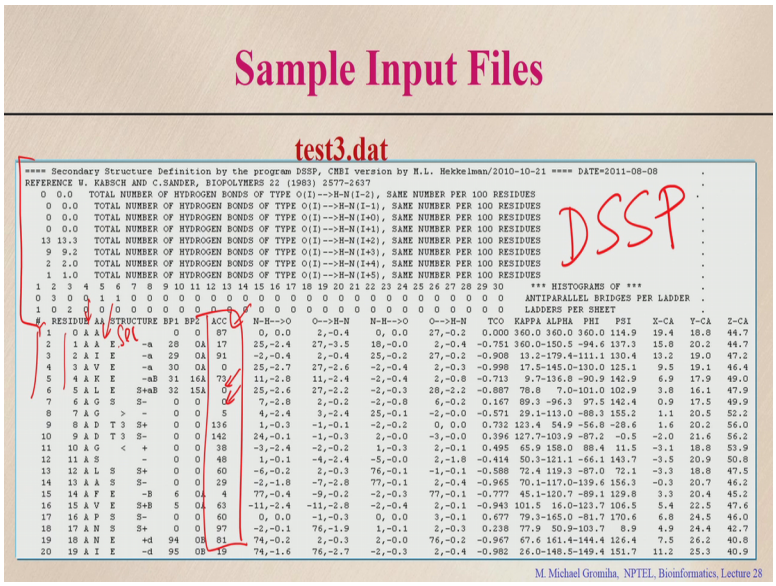
So, some of them are familiar to you already I showed several times as some of the new files, can you see this test two dot dat what is this?

Student: 3D.

It is kind of a it is a pdb file right. So, you can see the source, compound, sequence, right Helix, Atom and so on right this is a PDB file we can get the protein databank.

Now, in test one dat right here this is the output obtained from a program right we can see the various numbers this is the some scores and some ids right and some blank lines. So, these are the various types of information available in the test one dot, dat right we can use this data for manipulation.

(Refer Slide Time: 09:26)




```
==== Secondary Structure Definition by the program DSSP, CMBI version by M.L. Hekkelman/2010-10-21 ==== DATE=2011-08-08
REFERENCE W. KABSCH AND C.SANDER, BIOPOLYMERS 22 (1983) 2577-2637
  0  0.0   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I-2), SAME NUMBER PER 100 RESIDUES
  0  0.0   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I-1), SAME NUMBER PER 100 RESIDUES
  0  0.0   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I+0), SAME NUMBER PER 100 RESIDUES
  0  0.0   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I+1), SAME NUMBER PER 100 RESIDUES
 13 13.3   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I+2), SAME NUMBER PER 100 RESIDUES
  9  9.2   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I+3), SAME NUMBER PER 100 RESIDUES
  2  2.0   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I+4), SAME NUMBER PER 100 RESIDUES
  1  1.0   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I+5), SAME NUMBER PER 100 RESIDUES
  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30     *** HISTOGRAMS OF ***
  0  3  0  0  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0    ANTIPARALLEL BRIDGES PER LADDER
  1  0  2  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0    LADDERS PER SHEET
  #  RESIDUE AA STRUCTURE BP1 BP2  ACC     N-H-->O    O-->H-N    N-H-->O    O-->H-N    TCO  KAPPA ALPHA  PHI   PSI    X-CA   Y-CA   Z-CA
  1    0 A A               0   0   87       0, 0.0     2,-0.4     0, 0.0    27,-0.2   0.000 360.0 360.0 360.0 114.9   19.4   18.8   44.7
  2    1 A A  E     -a    28  0A   17      25,-2.4    27,-3.5    18,-0.0     2,-0.4  -0.751 360.0-150.5 -94.6 137.3   15.8   20.2   44.7
  3    2 A I  E     -a    29  0A   91      -2,-0.4     2,-0.4    25,-0.2    27,-0.2  -0.908  13.2-179.4-111.1 130.4   13.2   19.0   47.2
  4    3 A V  E     -a    30  0A    0      25,-2.7    27,-2.6    -2,-0.4     2,-0.3  -0.998  17.5-145.0-130.0 125.1    9.5   19.1   46.4
  5    4 A K  E     -aB   31 16A   73      11,-2.8    11,-2.4    -2,-0.4     2,-0.8  -0.713   9.7-136.8 -90.9 142.9    6.9   17.9   49.0
  6    5 A L  E     S+aB  32 15A    0      25,-2.6    27,-2.2    -2,-0.3    28,-2.2  -0.887  78.8   7.0-101.0 102.9    3.8   16.1   47.9
  7    6 A G  S     S-     0   0    0       7,-2.8     2,-0.2    -2,-0.8     6,-0.2   0.167  89.3 -96.3  97.5 142.4    0.9   17.5   49.9
  8    7 A G  >     -      0   0    5       4,-2.4     3,-2.4    25,-0.1    -2,-0.0  -0.571  29.1-113.0 -88.3 155.2    1.1   20.5   52.2
  9    8 A D  T 3   S+     0   0  136       1,-0.3    -1,-0.1    -2,-0.2     0, 0.0   0.732 123.4  54.9 -56.8 -28.6    1.6   20.2   56.0
 10    9 A D  T 3   S-     0   0  142      24,-0.1    -1,-0.3     2,-0.0    -3,-0.0   0.396 127.7-103.9 -87.2  -0.5   -2.0   21.6   56.2
 11   10 A G  <     +      0   0   38      -3,-2.4    -2,-0.2     1,-0.3     2,-0.1   0.495  65.9 158.0  88.4  11.5   -3.1   18.8   53.9
 12   11 A S              -  0   0   48       1,-0.1    -4,-2.4    -5,-0.0     2,-1.8  -0.414  50.3-121.1 -66.1 143.7   -3.5   20.9   50.8
 13   12 A L  S     S+     0   0   60      -6,-0.2     2,-0.3    76,-0.1    -1,-0.1  -0.588  72.4 119.3 -87.0  72.1   -3.3   18.8   47.5
 14   13 A A  S     S-     0   0   29      -2,-1.8    -7,-2.8    77,-0.1     2,-0.4  -0.965  70.1-117.0-139.6 156.3   -0.3   20.7   46.2
 15   14 A F  E     -B     6  0A    4      77,-0.4    -9,-0.2    -2,-0.3    77,-0.1  -0.777  45.1-120.7 -89.1 129.8    3.3   20.4   45.2
 16   15 A V  E     S+B    5  0A   63     -11,-2.4   -11,-2.8    -2,-0.4     2,-0.1  -0.943 101.5  16.0-123.7 106.5    5.4   22.5   47.6
 17   16 A P  S     S-     0   0   60       0, 0.0    -1,-0.3     0, 0.0     3,-0.1   0.677  79.3-165.0 -81.7 170.6    6.8   24.5   46.0
 18   17 A N  S     S+     0   0   97      -2,-0.1    76,-1.9     1,-0.1     2,-0.3   0.238  77.9  50.9-103.7   8.9    4.9   24.4   42.7
 19   18 A N  E     +d    94  0B   81      74,-0.2     2,-0.3     2,-0.0    76,-0.2  -0.967  67.6 161.4-144.4 126.4    7.5   26.2   40.8
 20   19 A I  E     -d    95  0B   19      74,-1.6    76,-2.7    -2,-0.3     2,-0.4  -0.982  26.0-148.5-149.4 151.7   11.2   25.3   40.9
```

Because this is test three dot dats are you familiar with this output.

Student: (Refer Time: 09:33).

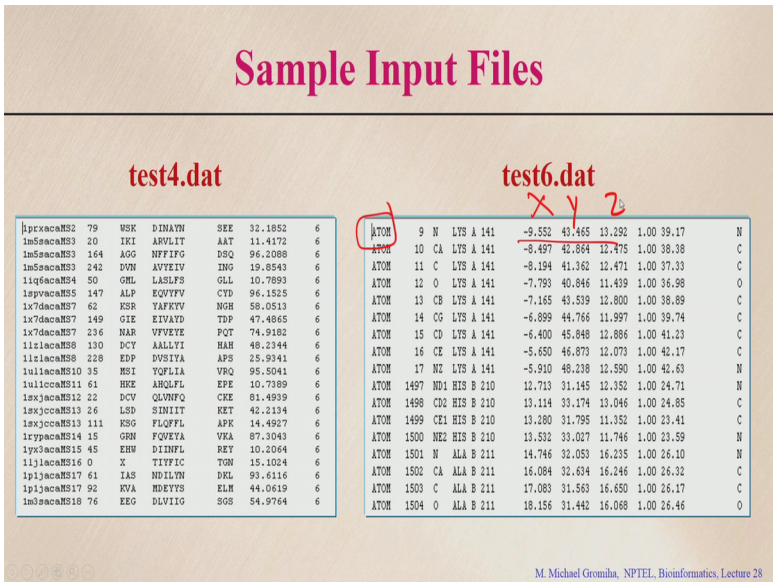
This DSSP output this you obtain from DSSP what is DSSP.

Student: Dictionary of secondary structure of proteins.

Dictionary of secondary structure of proteins right. So, you can see this one. So, here we have the comment lines right, how they identify the hydrogen bonds right and so on right. So, here from this it starts a data from one, here we give the residue name and the chain information here, and here we give the residue right and this is secondary structure, and we can see solvent accessibility right, what is accessibility? So, you know the how far the residue exposed or it is buried so, from this you can see that.

So, from this only we can easily identify the residues, which are at the surface or which are buried. So, because you have to search with this column right for example, if you need the completely buried say less than 5 angstrom square. So, see this column and see where we have less than 5 for example, here here right. So, you can use write one line awk code right to get the residues; residues are here right see here right which has the ASA of less than 5, easily you can see that right.

(Refer Slide Time: 10:46)

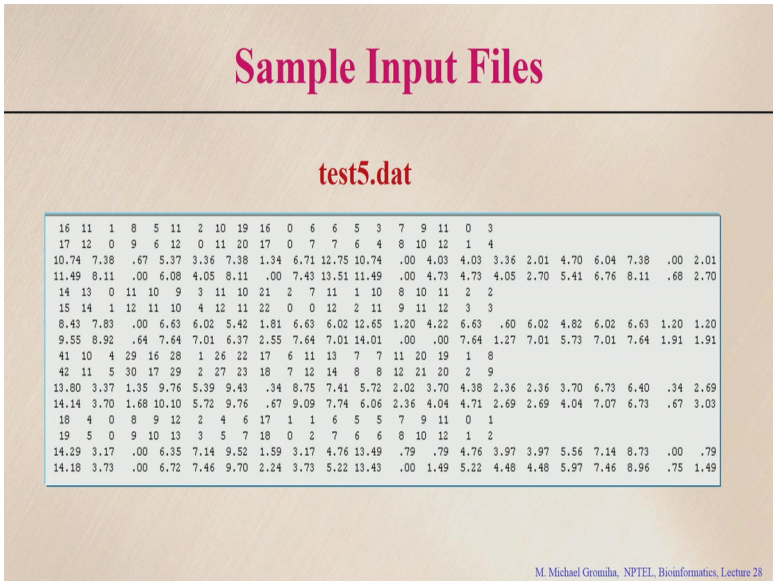


here I get few more data files, here I give one data, this is output for another program, test four and test 6 again it is a pdb file. So, what is this specific information in test 6?

Student: Atom records.

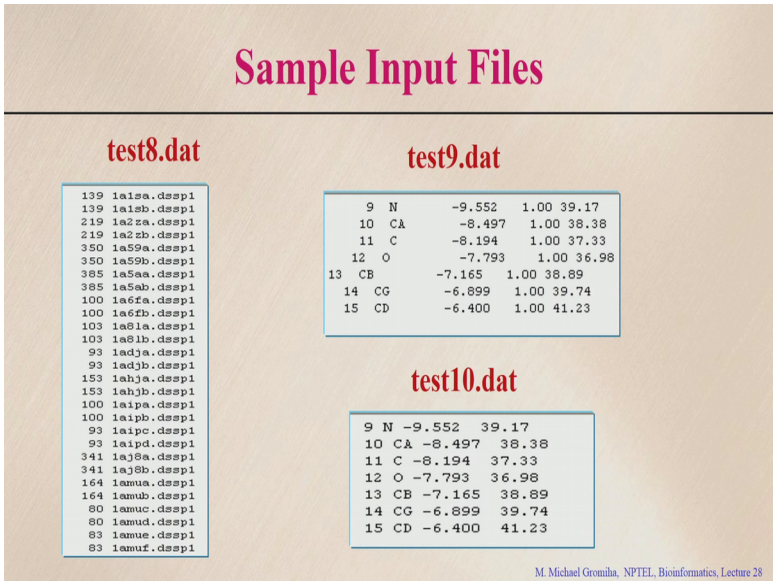
Atom record this one is atom record we have the coordinates right X Y Z coordinates right this is test 6, this is another text file.

(Refer Slide Time: 11:10)



This is the output obtained from the another program this one, you can also see how to use this test 5 also to do some kind of arithmetic calculations here.

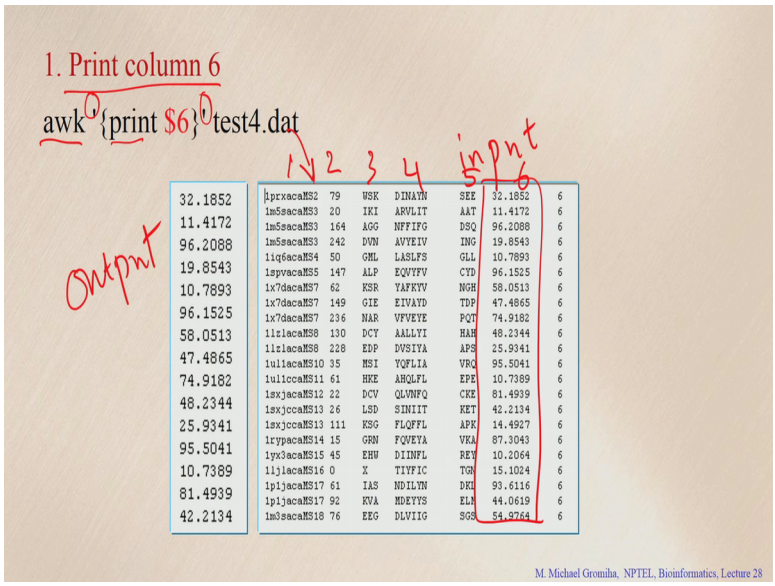
(Refer Slide Time: 11:21)



There is some list of input files this is input register input files. So, this is the number of lines and this is the file names. So, this is input file names.

Here this is the PDB coordinates we have only some information, not the complete information. Then this also this not in format right there are several misalignments in this particular test 9 and test 10. Now I will show you some examples. So, we have several input files and we desire to get the output files right based on some of our requirements right, in this case I will explain few commands using awk how to manipulate the data how to get the information what we need.

(Refer Slide Time: 12:06)



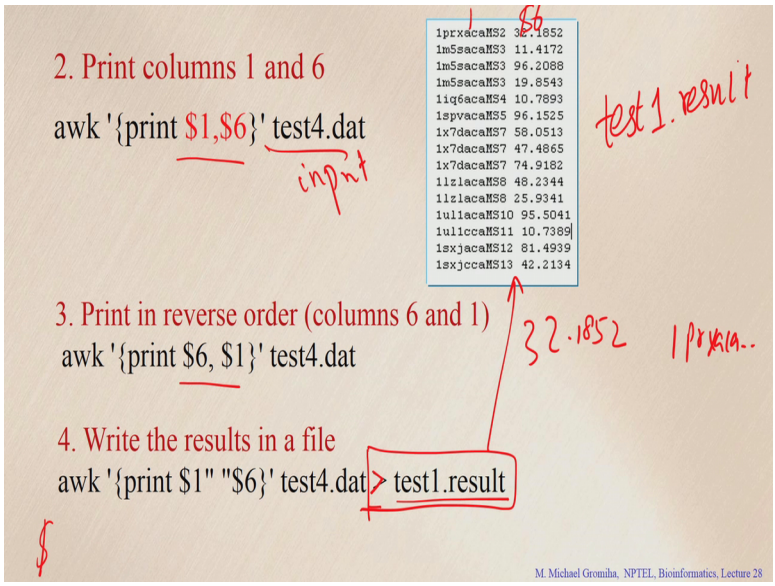
So, first one is, start from the simplest one, we have a file, this is the input file right this is the input file. Now the first question is print column 6, and which one is column 6? 1, 2, 3, 4, 5, 6 this is column 6 we need to print right. So, in this case do we need any pattern? We don't need any pattern because we are not looking for anything, just we want to print the column 6 because it contains some numbers; I want to see the numbers.

So, here you put the awk right and then what is the action want to do. So, there this bracket is important right. So, you have to give the action within quotes. So, you put the print what you want to do is you have print the 6. So, print this is the sixth column. So, you mention dollar 6 right then get the output. What is the output you will get if you work this on this file?

Student: sixth column

Right 6 column, like 32.8, 11.41 this is just you can see this output this you will get, right, that is fine for example, if you want to print two columns.

(Refer Slide Time: 13:29)



For example dollar one and dollar 6 right we need the; because this is only the numbers. So, I want to get the codes plus the numbers right. So, this case if you want to, how to do? Start awk and in the quotes you have to give the action, you have to print dollar 1 as well as dollar 6 right then you give the input file name, this is the input file name.

Then you will see this is the first column and this is the sixth column right you get the answers. If you see the input file this is your input file this is the 1, this is 6 and you get one and 6. The question is it possible to write in reverse order, right I do not want to get this 1 and 6, I want to write the results in 6 and 1, because I want to deal with 6 first.

Is it possible to write the data in the reverse order it is possible how to do this in this case? You can see the order. So, here this will follow the same order what you give an action. If you ask to write a first column we will write first column, 6 column 6 column if you want to take several columns what are the order you give accordingly you can write. See if you want to first and 6 means you will get first and 6. If you write 6 and 1 what will be the result? Yeah this will be like this.

32.1852 then 1 p r x a c a like this right you can get the answer. Now here this is the screen. So, if you write just the input files right and if you give the awk command you can see the data on the screen right for example, if we want to write the answer in a file right, in this case what to do right we have to give the greater than symbol; that means,

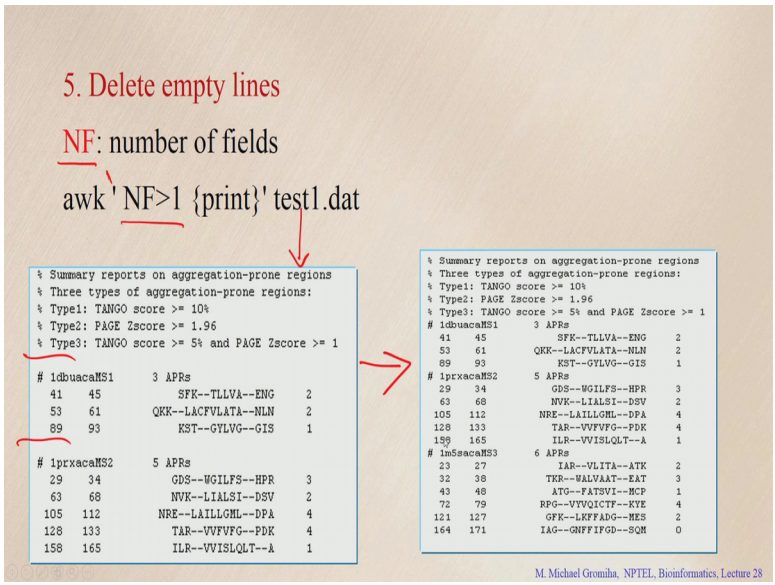
the result obtained from this command will go to this file right this is the extra we give this is the same.

Awk print 1 comma 6 right and you need to specify where you want to write. So, you could put the greater than symbol, this is the output will go to the file what do you specify right. So, you can see test one dot result right. If you do it where can we see it on the screen now or only on the file it is on the file.

So, if you do like this then the next will come with this dollar symbol for the next command. Because if you open this file right then you can see if you open test one dot result there you can see this one comma 6 means this result right. This is stored in test one dot result right that is fine. So, for any operations you can either see your results on the screen or you can write in a file.

So, I have a question if you have several empty lines for example.

(Refer Slide Time: 16:33)



If this is the data there are several empty lines right, in this case how to eliminate the empty lines. So, I need the file without empty lines right. So, here we use this NF, NF is number of fields. what is the meaning of number of fields?

Student: Number of columns.

Number of columns. How many fields we have. So, if you want to eliminate the free lines. So, at least this contain only one column right there is one field. So, in this case you put NF is greater than 1 right, you can see the codes it started from this your conditions till your action right. So, here the condition is NF is greater than 1. What is the meaning of NF greater than 1?

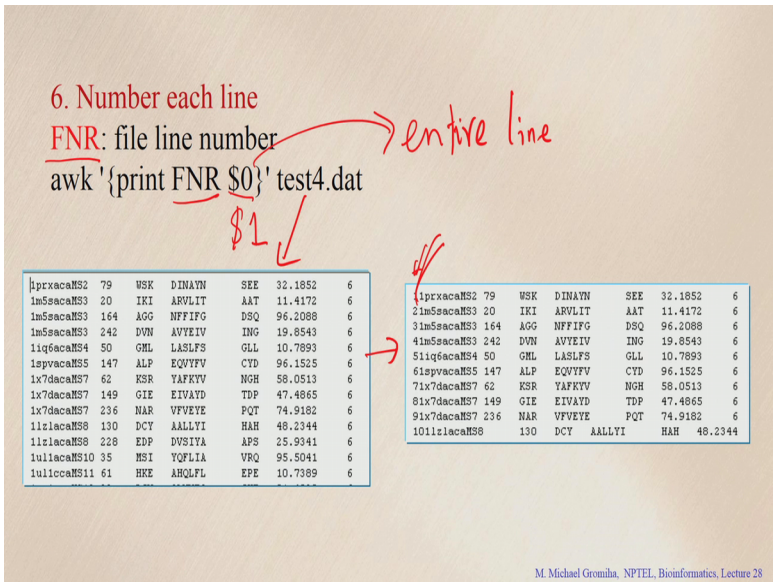
Student: number of fields is

The number of fields will be more than 1 that should be more if it is more than one then you print right, but here you will get the result in the screen or in a file.

Student: Screen.

On the screen right you can see it on the screen. So, here if you do if you work on this one right then you will get this result; these empty lines are eliminated right all others are fine. Type three these empty lines eliminated and this empty line eliminated right this is fine. So, now, next question is I have a file.

(Refer Slide Time: 17:52)



I want to do number in each lines right. So, in this case, what we have to do. So, here we have this command FNR, FNR gives file line number right for each fields for each line. This will give a number.

See if you want to number all the, consider all the columns and print everything right. So, print FNR, FNR means give the file number and dollar 0. What is the meaning of dollar 0?

Student: Entire line.

Entire line.

So, if you write dollar one what will happen.

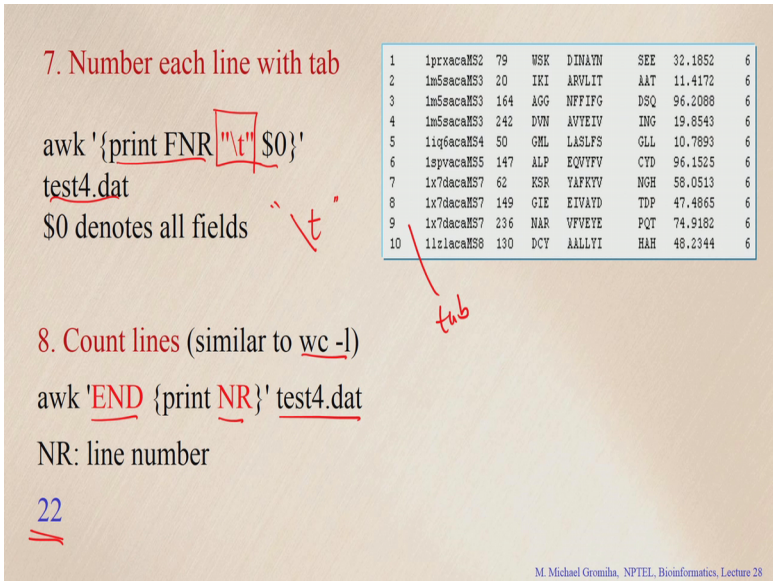
Student: First columns.

This will give only the first column if it is dollar 0 so; that means, the entire. So, you write the entire line right. So, in this case if you see this one whether in the output, this is the input right. So, get the output. So, did you see it added the line the line numbers?

Student: Yes.

Yes right we can see the line numbers, but this is merged with this first column right. So, in this case it is very difficult to see where the numbers are. So, in this case it is better if you add a tab or add some space between these two, because there are no space. So, it is merged right. So, in this case what can we do is it possible to add a tab in between these two? Yes it is we can add it you can have this command.

(Refer Slide Time: 19:24)



The same print FNR dollar 0 test four dot dat right, but here we added another information that is tab, that is backslash t right you put in the quotes right. If you do this then it will add lines right with a tab. Now can you see this this result and the previous result. Here there is no space, but here you can see this is a tab right. So, we can also give space right when we are adding numbers ok.

Now, the question is you can also count lines, if you take many documents in a file. So, how many lines ? In the Unix we give the command what is the command we use to count the line number of lines?

Student: wc -l.

wc minus l just we give wc minus l this will give you the lines right. So, here also you can give awk, you go from to the end. What is NR?

Student: line number

Number of lines. So, they are line numbers right.

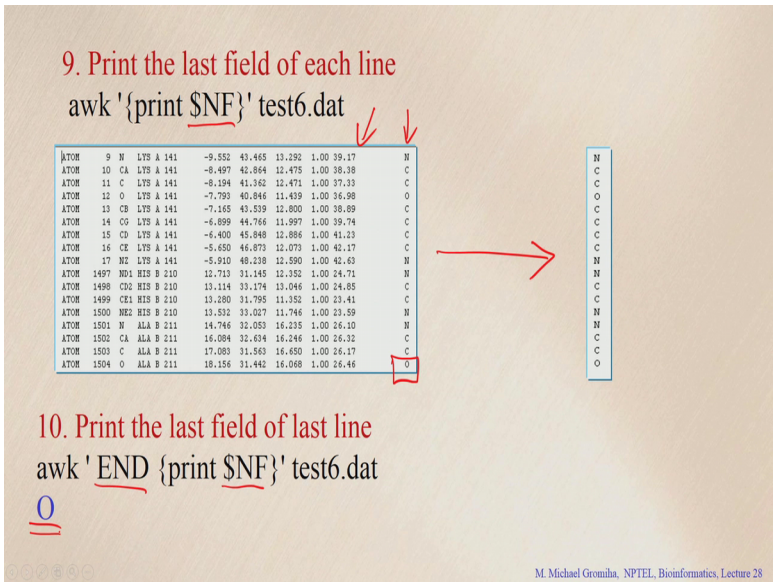
Student: Number of records.

So, you can.

Student: Number of records.

Yeah number of records. So, you can see NR right. So, how many lines say if you go to this particular file test 4 dot dat the input file. So, you give the line number. So, count the all the lines. So, it will give total 22 records say 22 lines in this particular file. So, it is similar to wc minus l in the Unix right.

(Refer Slide Time: 20:55)

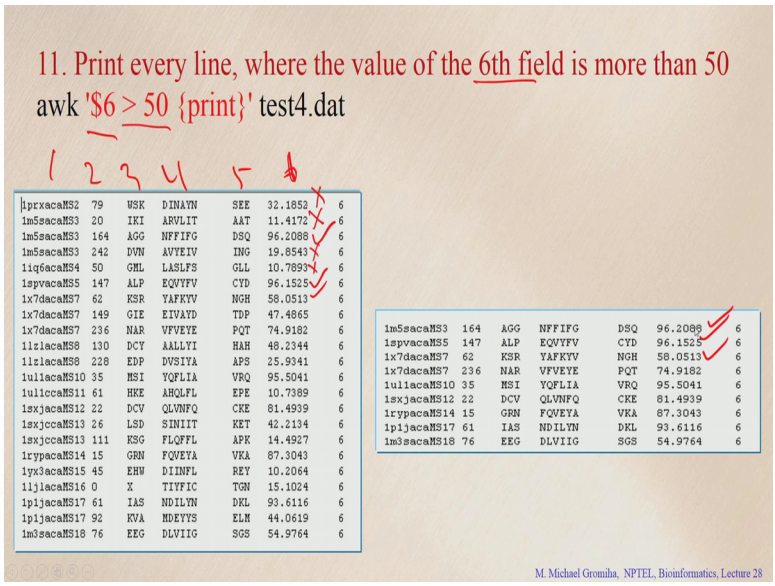


So, now here the next question is print the last field of each line. So, we have several fields. If you know this is you require 5 field or 6 field, just you write print dollar number of fields 5 or field 6 right. If it is a last field then you put dollar NF right, In this file, so, what is the last field? This is the last field right then if you do this you will get the result. So, just it is simple awk command, awk print dollar NF this input file test 6 dot dat. So, we get the last field of each line.

Then also it is also possible the last field of last line right. Just now we saw about the total number of lines. So, you put the end right when you give this line number. So, same you combine. So, this is the last field this is the end and they print dollar NF right this is the last field or last line that is this is O. So, you can see the result over here right.

So, likewise you can manipulate right any data file right as we require, then I will show some of the important information where we can really manipulate the output files.

(Refer Slide Time: 22:06)



So, here I put one condition print every line where the value of the 6 field is more than 50 right. So, how to do this? So, you can see the 6 field right, I have 6 field, more than 50, it is a more than 50 then you print if I just print.

What it will do? It will print, the entire, either print dollar 0 or you just give print. So, you can give the entire things if it is more than 50. So, if it is more than 50 no right this is more than 50 dollar this is 1, 2, 3, 4, 5, 6 right this is more than second one more than 50 no third one yes then again this no no this 6 right.

So, you can see the same way. So, here you can see this is 96 is present here and again the next stand is present here, and 58 is present here. So, you can see the values right which is more than 50. Now if I have this example I have the, I give the example. So, I want to see how many amino acids are in the buried region. So, what you will do?

If you have any pdbid you can if want to see the number of residues which are in the buried. So, you can run the DSSP. So, you will get this output from this is it possible to see get the number of residues, which are completely buried for example, 0 without touching any of these file is it possible to do we can do, what we have to do first?

Student: Select only records.

Yeah select only the records which number 1, 2, 3, 4, 5, 6, 7, 8, 9 right take this field and then see you have to write the records if this record is more than 50 and second what we

have to do? We can do the number of fields likewise if we make two commands and you can easily get the number of residues, which are at the buried because if you take the condition of 0 right. So, these are the two commands we have to do.

First we need to right check the file right if the column 9 right which is more than 50 right, and then this will give you the residues which are having the value of 0. Now if it is buried means you have to put that less than 1 or equal to 0 right and then you can use these command right this is the command to see the NF minus l right. So, then we will see how many records.

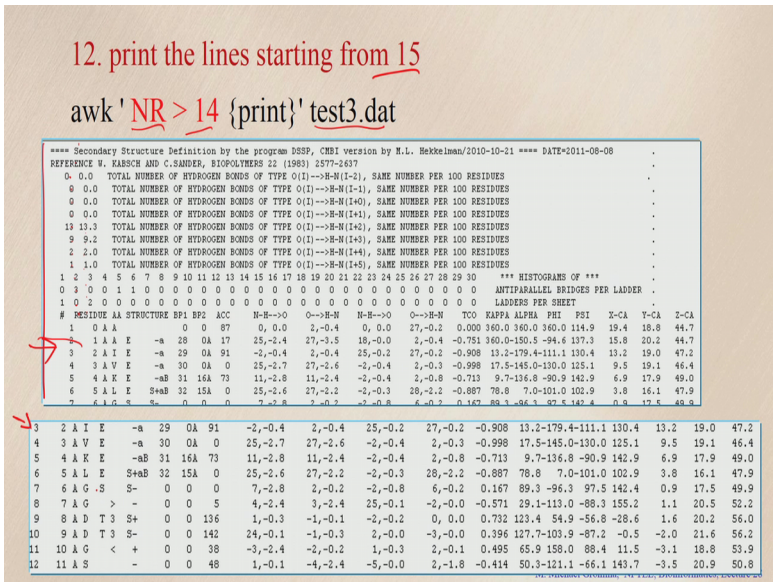
Now is it possible to identify how many residues which are in the exposed region with a condition of 75 angstrom square.

Student: Yes.

Right. So, first what we have to do? We have to put this column number which is greater than 75 then you print right and you print this two output, right this output will be the input of this the next command. So, here you can give the command, this is the output file you give here and this will give you a total number of records right. So, in this it will tell you without looking into the PDB file or DSSP file right we can see how may residues right, which are buried how many residues are which are in the exposed right.

Just first take the PDB file, run DSSP right then give the awk command. So, with field which contains the value of more than 75, then you find the line number, totally how many lines right. Without manipulating directly anything, we can easily get all the information. In this case also you can see how many alanines which are exposed, which are in the buried right likewise you can have different files we can manipulate the files, easily you can get all the informations right without editing any of these files.

(Refer Slide Time: 26:30)



So, the next question print the lines starting from 15 right. This is your DSSP file right. So, here these are your command the information right. So, actual result starts from here right. If you want only these data as the input to the another program because this they do not need all these unnecessary data. So, in this case if you want to have the lines right which is starting from15.

So, here you can, what is NR.

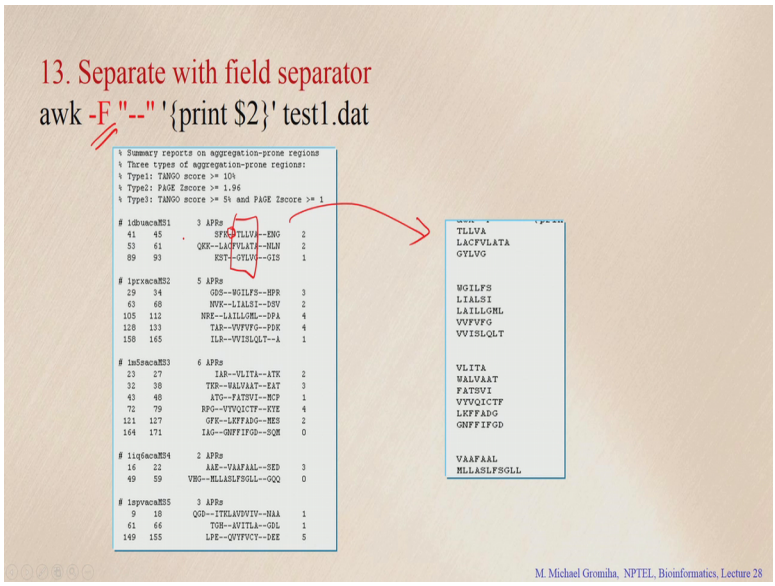
Student: Records.

Records right. So, in this case line number this is more than 14, because we need from 15 right then you print from this input file. So, then you will eliminate all these things 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 1,2, 13, 14.

So, then you can start this is 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14 right. So, then we can get from this line right this line.

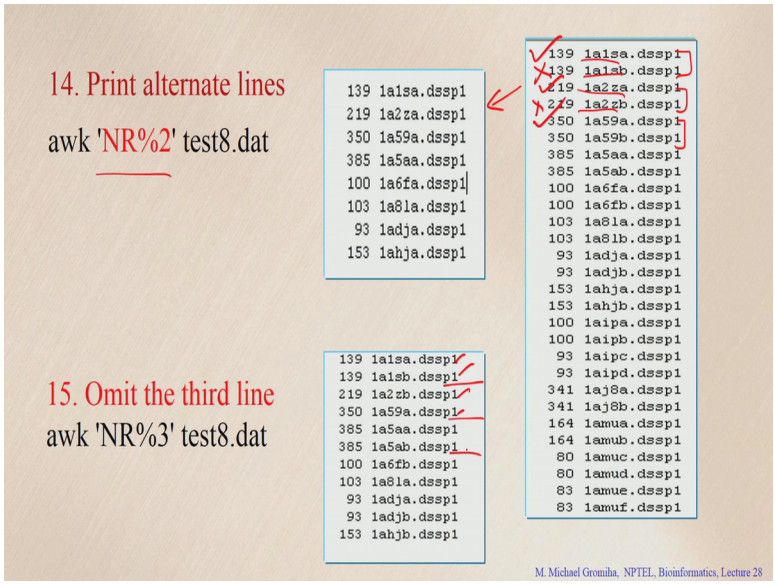
So, you can give any number according to the number you give. So, you will get the output so; that means, you delete the line numbers until 14 lines right this is you can check or you can get these a files. So, you can do with any numbers right 15 or 16 whatever you like. So, you can do that.

(Refer Slide Time: 28:00)



So, then the next question is you can separate the field separator for example, if there is any field separator this is the field separator, here you can see this one whether we can eliminate you can do this separate these things. So, if this is the case right, dollar 2. So, you can print the dollar two like test one dot dat. So, this is the test one right now here you can see the field separator, this is the 'F' for field separator and we can get this to here right the middle ones, separate these two files right.

(Refer Slide Time: 28:29)



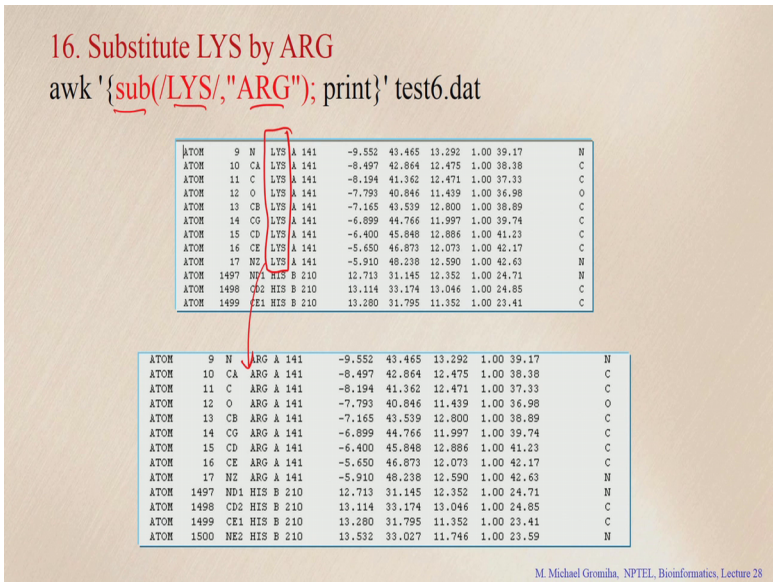
In this case you can also do other lot of other manipulations for example; you can print the alternate lines.

In this case if you see this input file. So, this is the you can see the same for example, one a one s, one a one s, one a two z, one a two z. So, each these two are the same right, only difference is a and b. So, I want to see the unique wants, in this case I like to delete the alternate lines because that input is necessary for the another program. So, if you want to print alternate lines, you can omit the second one using this equation like the NR dollar this is NR percent two; that means, it will cut the second one and finally, you get this result right

So, this is here this is omitted, this is here, this is omitted this here right you can get that one. So, if you want to go to the third line right then you put the 3. So, first two is present here and the third one this 2 one 9 is omitted and the again you start from this two right and the third one is omitted here you can see the omitted ones right you can do for any lines if you want to omit.

Now, if you want to do any substitutions for example, if you want any mutations. So, you can it is possible to substitute any text for example, using this command this sub. So, what do you want to substitute?

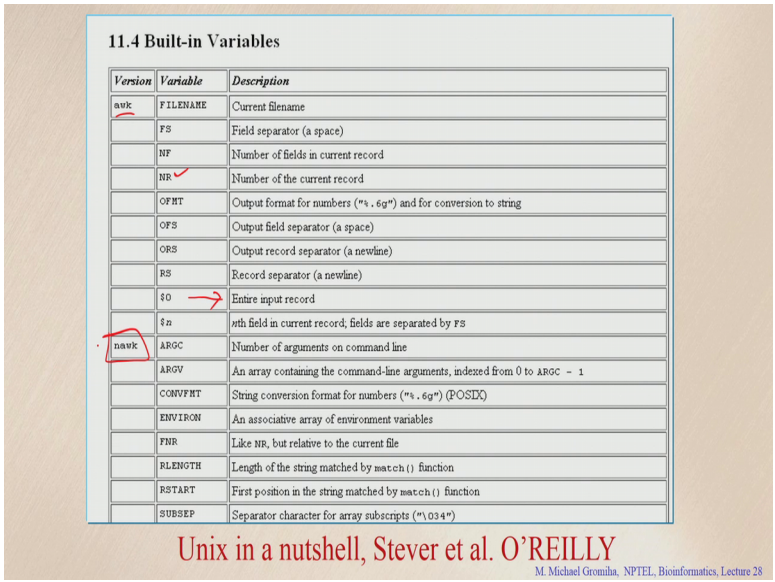
(Refer Slide Time: 29:57)



Here you want to substitute LYS to ARG right how to substitute this lysine to arginine? So, here is a record for the lysine right. So, the change it to arginine by giving this command that is awk you get this is the pattern action statement sub is your substitution from LYS. So, in this case you have to put the slash, wherever the exactly you get the LYS you will change to arginine.

So, here if you see this one LYS into arginine this one example, but you can do the manipulation for the different files at the different places or if you want to make any changes, like kind of this replace command ok.

(Refer Slide Time: 30:40)



11.4 Built-in Variables

Version	*Variable*	*Description*
awk	FILENAME	Current filename
	FS	Field separator (a space)
	NF	Number of fields in current record
	NR	Number of the current record
	OFMT	Output format for numbers ("%.6g") and for conversion to string
	OFS	Output field separator (a space)
	ORS	Output record separator (a newline)
	RS	Record separator (a newline)
	$0	Entire input record
	$*n*	*n*th field in current record; fields are separated by FS
nawk	ARGC	Number of arguments on command line
	ARGV	An array containing the command-line arguments, indexed from 0 to ARGC - 1
	CONVFMT	String conversion format for numbers ("%.6g") (POSIX)
	ENVIRON	An associative array of environment variables
	FNR	Like NR, but relative to the current file
	RLENGTH	Length of the string matched by match() function
	RSTART	First position in the string matched by match() function
	SUBSEP	Separator character for array subscripts ("\034")

Unix in a nutshell, Stever et al. O'REILLY

M. Michael Gromiha, NPTEL, Bioinformatics, Lecture 28

Now, in this awk there is several built in function available. So, I explained few of them. So, there are various built in functions available in awk. So, for example, FS is the field separator, I discussed earlier what is NF.

Student: Number of fields.

Number of fields in the current record, NR number of current record likewise there are various other options like record separator, this is dollar 0 is for the entire input record. Dollar n is the nth field in the current record right. So, likewise you can see this FNR like the NR, but related to the current file you see any of these variables and get the description right and accordingly you can manipulate these files using these awk command.

So, again this some of this available in awk and you can see some more information right we can also use nawk right for a more information. So, now, we go to next step now first one is we discussed about the substitution and you are printing the some lines some columns right and doing some sort of a file manipulations.