Next Generation Sequencing Technologies: Data Analysis and Applications

Basic Shell Commands

Dr. Riddhiman Dhar, **Department of Biotechnology**

Indian Institute of Technology, Kharagpur

Good day, everyone. Welcome to the course on Next Generation Sequencing Technology Data Analysis and Applications. We have been discussing basic shell commands as part of the first hands-on, and we will discuss a few more commands today that will be useful for NGS data analysis. So, these are the topics that we will be covering today. So, we will discuss shell commands for file compression and decompression. So, the NGS data files, FASTQ files, for example, come with different compressions sometimes.

So, you may have to decompress those files. In addition, you will use many tools; they also come in certain compressions. So, we would have to decompress them and install them. And the second topic that we will be discussing today is the search expression.

Now the files that we deal with are really huge files, right? So, you cannot just open them and search for a specific read sequence or read ID if there is a need. So, we would have to use these command-line tools, which will search these files very efficiently. So, these are the keywords: gzip, tar, and grep. So, these are the commands that we will encounter in today's class.

So, let us start with the file compression part. So, the first file compression that we will use is called zip. So, this is a common file compression that is also present in Windows; you must be familiar with this compression. In Linux, what do you do if we have to create a zip file using the command line? And similarly, what do you do if we have to decompress the file?

So, we will discuss that one step by step. So, this is the command that we use. So, the first is zip, creating this zip file. So, how do you create this zip file? So, this is the command zip. So, the commands are shown in green right next to that part of the command, and then we have the file name that you want to create after zipping.

So, this is file 1 dot zip, followed by the file name from which you want to create this zip file. So, file 1 is the file from which we want to create this zip file. So, this is the file name that is given, okay? So, this is a slightly different notation, right? So, the zip file name would come before the actual file, okay?

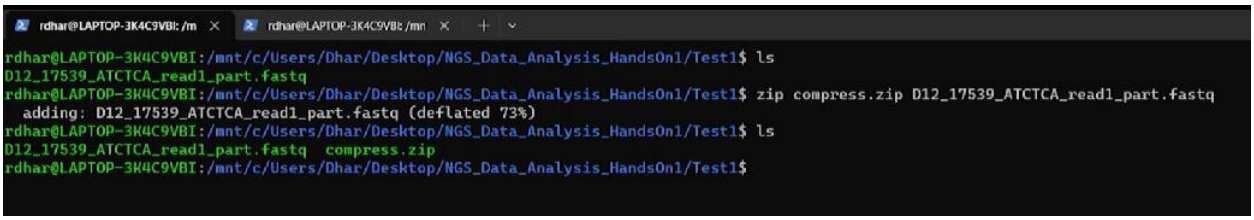
Now, similarly, if you want to compress a folder itself containing multiple files, this is how you

should write it. So, you have this; you would have to add this option, zip minus, ok? So, minus r means recursively. So, it will compress all the files and all the contents of that folder into that folder. And then followed by the folder name from which you want to create this folder, ok?

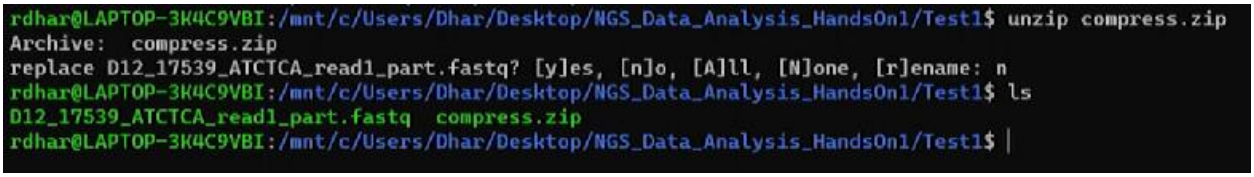
So, we will see in a moment how this works. So, how do you unzip a file? So, for unzipping files, you just simply say unzip and file name, the zip file name from which you want to create this decompressed file. So, this is how the thing will work, right? So, let us try this out on the command line, right? So, we will try that out, and as part of the hands-on, you can try in parallel, ok?

So, if you remember some of the commands that we used last time, So, one of the commands is right. So, this lists the files that are in this directory, ok? So, this is the file name D12, some specific fastq file. This is the fastq file we are working with right? So, let us zip this file, right?

So, how do you compress, right? So, the command is zip compress dot zip, and then the file name is right. So, the file name is D12, which is the full file name, right? This fastq file we want to compress into a zip file, so we will call this new file compressed dot zip. And this is done now, right, and as you can see, this zip file is compressed dot zip.



Now, if you want to again unzip this right. So, we simply write unzip compress dot zip right, and it will say replace we do not want to replace, or if you want to replace, you can say yes, and if you do not want to replace, you can say no, just right. So, you just have this right, ok? So, this is how this zip and unzip works, ok? So, this is a very common compression that we also use in Windows.



Then we have other compressions; these are actually Linux-specific compressions, ok, and these are more efficient; you can actually control the level of compression as well, ok. So, the command for compression that we use most commonly is called gzip compression. So, you will see this in the fastq file; sometimes they will also come with this gzip compression, and we will see this dot gz at the end of the file names. So, how do you compress this gzip file name? Now, if you do gzip file name, what will happen is that the program will simply take this file name and compress it, and the original file will be lost. You will not have that original file anymore.

Now, if you do not want to lose that original file, these are the options that you have, ok? You can say gzip minus c file name to file name dot cz, right? So, their format is slightly different from zip here; you want to create this right gzip minus c file name to file name dot cz, or you can simply say gzip minus k file name. So, minus K would suggest I keep the original file. So, this is how this whole thing will work.

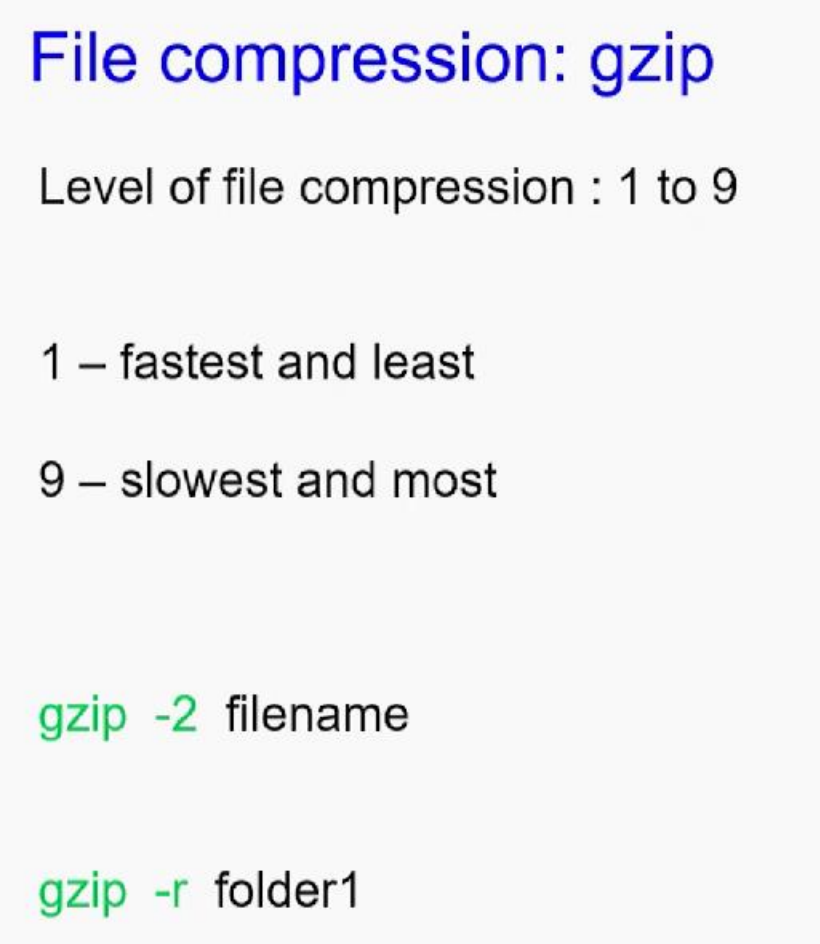
File compression: gzip

```
gzip  filename
```

```
gzip  -c  filename > filename.gz
```

or

```
gzip  -k  filename
```



The other thing is that you can control the level of compression with gzip. The levels are 1 to 9. 1 is the least compressed, which means it would take more space on the hard drive and it is the fastest. It does not need to compress too much. So, it will be very quick, whereas 9 is the slowest and the most compressed. So, how do you actually control this? How do you control this level of compression? You say gzip minus 2 file names, ok?

So, these are the options. So, you probably have an idea of how to use these shell commands. You have the commands and following those commands, you can sometimes use certain options. So, this is very common across all Linux commands, ok? In case you want to compress a folder, you have to again give gzip minus r folder 1 right. This is again the same as what we have seen earlier: if you want to compress a folder, you just say gzip minus r.

```
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ gzip D12_17539_ATCTCA_read1_part.fastq
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ls
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ls
D12_17539_ATCTCA_read1_part.fastq.gz  compress.zip
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$
```

So, let us try this out with gzip in our command line, ok? And this is what we will do with Gzip; we just keep the file name, ok? And this is done right, and what you see here is this file, now the original file, D12. This g1 part dot fastq dot gz is okay. Maybe you can zoom in a bit. So, that is easier for you to see, right?

So, let us do the LS again, and you see this right. This original fastq file is now fastq dot gz ok. So, there is this Gzip compression, okay? Now, how do you undo this right? So, ok, so we have not talked about that yet, right? How do you actually undo this gzip? The other thing we talked about is whether you want to create or keep the original.

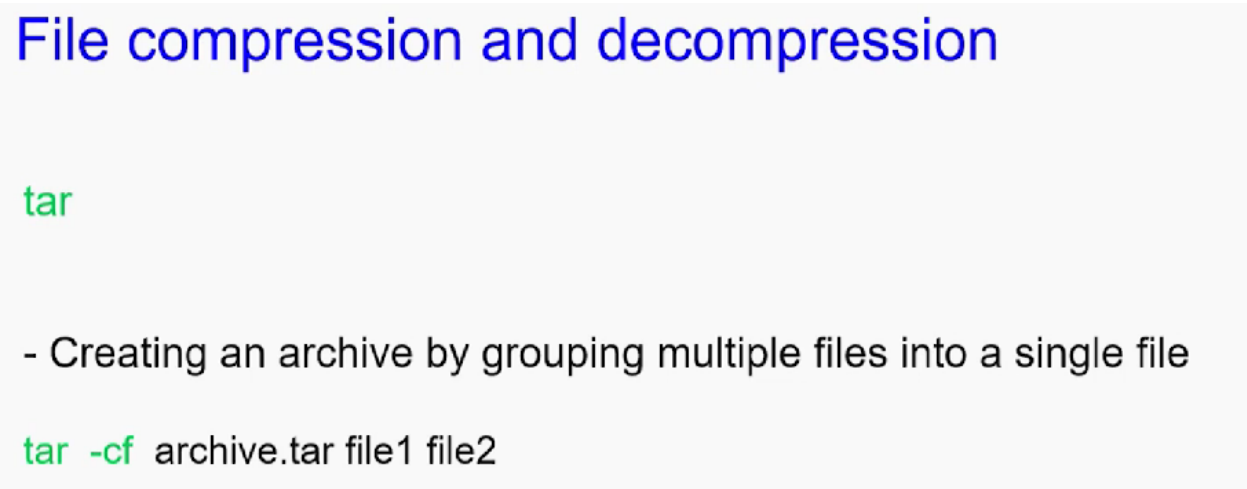
So, first, we have to undo this right; we have to decompress. So, decompression works like this. So, these are the two commands that we use for decompression. One is this gzip minus d file name, dot gz. Right, you give the gz file and say gzip minus d, or you can simply say g unzip ok. So, let us try that g unzip right, and then we will try the other commands that we have discussed with gzip. So, gzip minus d is right, and the gzip file is okay.

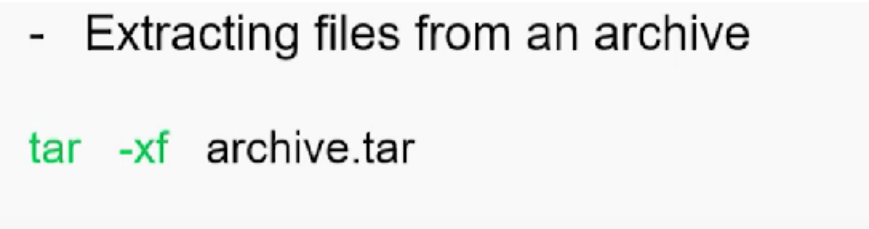
```
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ gzip -d D12_17539_ATCTCA_read1_part.fastq.g
z
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ls
D12_17539_ATCTCA_read1_part.fastq  compress.zip
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$
```

```
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ gzip -k D12_17539_ATCTCA_read1_part.fastq
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ls
D12_17539_ATCTCA_read1_part.fastq  D12_17539_ATCTCA_read1_part.fastq.gz  compress.zip
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$
```

And it is done right, and you can check right just by saying list, you can simply check, and we are back to the original dot-fastq format, ok? And now let us try this gzip minus c dot minus k right. Keep the original file right. So, gzip minus k is right, ok, and it is done right, and you see now there are two files right. So, the original file, the fastq file, is there, and then we have this fastq dot gz right. So, you get an idea of how these commands actually work, all right?

So, let us move on to the next compression, which is also quite common in Linux systems, and you will see many software tools that will come with this compression. So, this is the command we use; it is called tar, right? So, what it does is create an archive by grouping multiple files into a single file, and it is a really elaborate command. It has a lot of options, and you can control a lot of things. So, if you want to compress multiple files, what you will do is take this file 1 file 2 right. Look at the format of this command; we have tar minus cf right archive dot tar. So, the file that you want to create should come first, and the extension is dot-tar.





- Extracting files from an archive to a specific directory

```
tar -xf archive.tar -C directory
```

- Creating a compressed gzipped archive

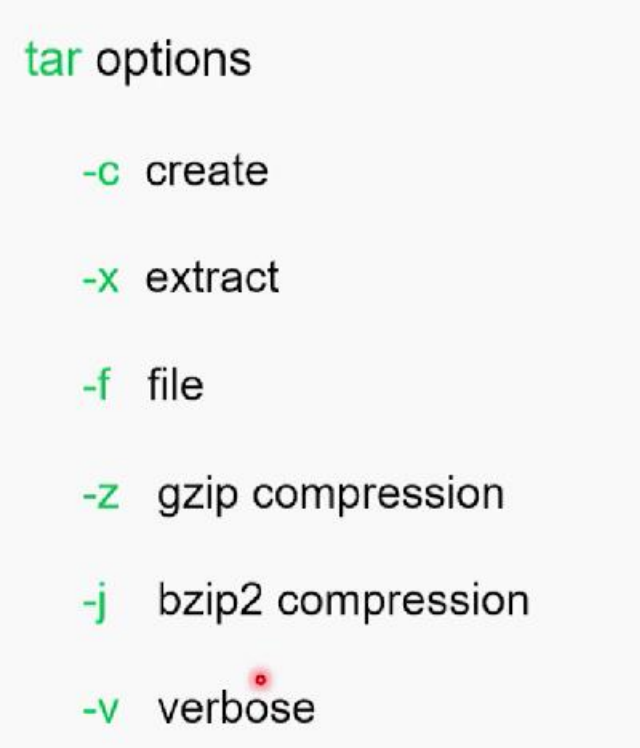
```
tar -czf archive.tar.gz file1 file2
```

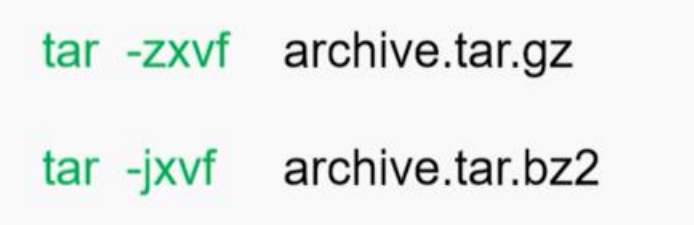
- Extracting files from a gzipped archive

```
tar -xf archive.tar.gz
```

or

```
tar -zxf archive.tar.gz
```





And the two file names or three file names are multiple file names. As many files as you want to compress, you just put those file names one after another. So, in this example, you want to compress files 1 and 2. So, we create this archive dot tar from files 1 and 2, ok? If you want to extract, okay. So, if you want to create, we use this minus c, right?

So, f means file. So, minus cf. right. So, create this file x means extract, ok? So, that is what that is why the option minus xf ok. So, you have created a tar file, and you want to extract it, so you will write tar minus xf archive dot tar, ok? If you want to extract these files into specific data, let us say you have hundreds of files. You know there will be many files in this archive, right?

So, what you do is specify a directory instead of the current working directory. You want to specify

another directory where this extraction process will happen, and all the files will be extracted into that directory. This is the option you will give minus the C directory. Now, if you want to create a gzip version of this archive, ok. So, we have talked about this, right? So, here we are creating this archive tar, and on top of that, we are applying this gzip compression.

So, you see this dot tar dot czip, right? So, this is something you will see for many tools, many softwares, and sometimes many data files. If you want to achieve really good compression from multiple files, ok. So, this is the command you will use: tar minus czip f right archive dot tar dot czip right. So, you see these two compressions: dot tar dot czip. So, you use, you know, this is a compressed gzip file, and then the file names that you want to create the archive from.

Now, if you want to extract this, So, again, you can simply use tar minus xf. Actually, tar knows if it sees dot tar dot czip; it automatically knows this is gzip ok. So, you do not have to again specify that this is a gzip file. So, use this minus xzf or minus zxf, but if you want, you can also use minus zxf. You are explicitly mentioning this right minus zxf here, ok? So, these are the two options that you have:.

You actually have many more options here. So, some of them we have talked about. So, minus c is create, minus x is extract, minus f is file, and minus z is gzip compression. So, if you want to create this gzip compressed archive, we use the minus z option, and there is another type of compression, which is called bzip compression in Linux. So, for that, we use this minus option, ok? And sometimes we also use this minus v option, which means the program or that command will tell us what it is actually doing.

It will kind of give a step-by-step process that is going on. So, let us try this right. These are some of the examples that we have right, and you can merge this command. So, if you are working with this tar dot gz file you want to extract, you will say tar minus zxvf ok. This is something that will be very useful when we want to install tools. You will see that many of them will come in this tar-dot-gz format. If you want to extract this tar dot bz2 file, you will use this minus zxvf file.

This is something else again. If you see this bz2 compression, this is something you will have to

use, ok? So, let us try this out, okay? Let us try this in our command line and let us get a field right. Of course, we will not try all the commands, but at least if we try some of the options, you will get a better understanding, right? So, if you want to create this archive, what we have to do is first get another file. We have this dot-fasq, so let us get another file.

And we can utilize a command that we discussed last time, if you remember. So, if you remember, we used this pwd command, which is the present working directory, and inside this, inside this test1, we were inside this test1, right? If you go out of test 1, you may not remember, right? If we go out using the cd dot dot command, there is this genome data analysis folder, ok? So, inside this folder again, there are multiple files. So, what we will do is copy another file so that we can demonstrate this compression by creating this archive, etcetera, using Tar.

So, let us do that. So, what is the command for copying? So, it is actually CP, right? So, cp the path name where this file is right. This is that folder inside which we have this other file, and let us copy the other file that is there, and the fasq file is there, and we copy this to the Test1 directory, ok? So, this is the format we discussed last time: CP genome data analysis, the file name, and the destination directory. If the destination directory is the current directory, we can simply say dot ok. We can just leave a dot, so the command will understand that the destination directory is the current directory, ok, but our destination directory is Test1.

So, we will move into Test1, and we will see that this file is there now, ok? So, again, we go inside this Test1, and we can check that these files are there, ok? Now, we see these two files as two fasq files, and what we want to do is create an archive out of these fasq files. So, these two fasq files let us compress them, and we use this star command minus c right to create this compression.

We want to z we want to compress with z zip right. So, z zip compressed and f, and maybe we, if you want to see right what the program is doing, we want to also use this option here right what the command is doing. It will tell us what is doing okay. So, the command again is you remember right compressed start dot sorry start dot c z right. The file name will have to give the archive that we want to create first, followed by the two file names.

```
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ls
D12_17539_ATCTCA_read1_part.fastq  D12_17539_ATCTCA_read1_part.fastq.gz  D12_17539_ATCTCA_read2_part.fastq  compress.zip
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ tar -czvf Compress.tar.gz D12_17539_ATCTCA_
read1_part.fastq D12_17539_ATCTCA_read2_part.fastq
D12_17539_ATCTCA_read1_part.fastq
D12_17539_ATCTCA_read2_part.fastq
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$
```

```
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ls
Compress.tar.gz                  D12_17539_ATCTCA_read1_part.fastq.gz  compress.zip
D12_17539_ATCTCA_read1_part.fastq  D12_17539_ATCTCA_read2_part.fastq
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$
```

We want to compress the fasq files. Right, two fasq files are okay. So, these two files are there, okay? So, let us run this command; it will tell us what is doing okay. So, it just did this by reading these two files, and it has created this compression. So, let us check if the compression is there. If the archive is created, you see this compress start dot.

So, this has been created. We can check the sizes right of these compressions, and you can compare right how tar, how z zip, and how zip actually compare in terms of storage space, and you can see which one is the most efficient. This is something you can try; we will see the command for how you can check this bit later, alright? So, now we want to unzip or decompress right this start dot gz. What is the command for this? So, this will be star minus z x v f, right? x is for extract right; you can simply say x v f right instead of saying z x v f, but we can use this and the file name right. So, let us unzip this. Maybe we want to do this in another directory, right?

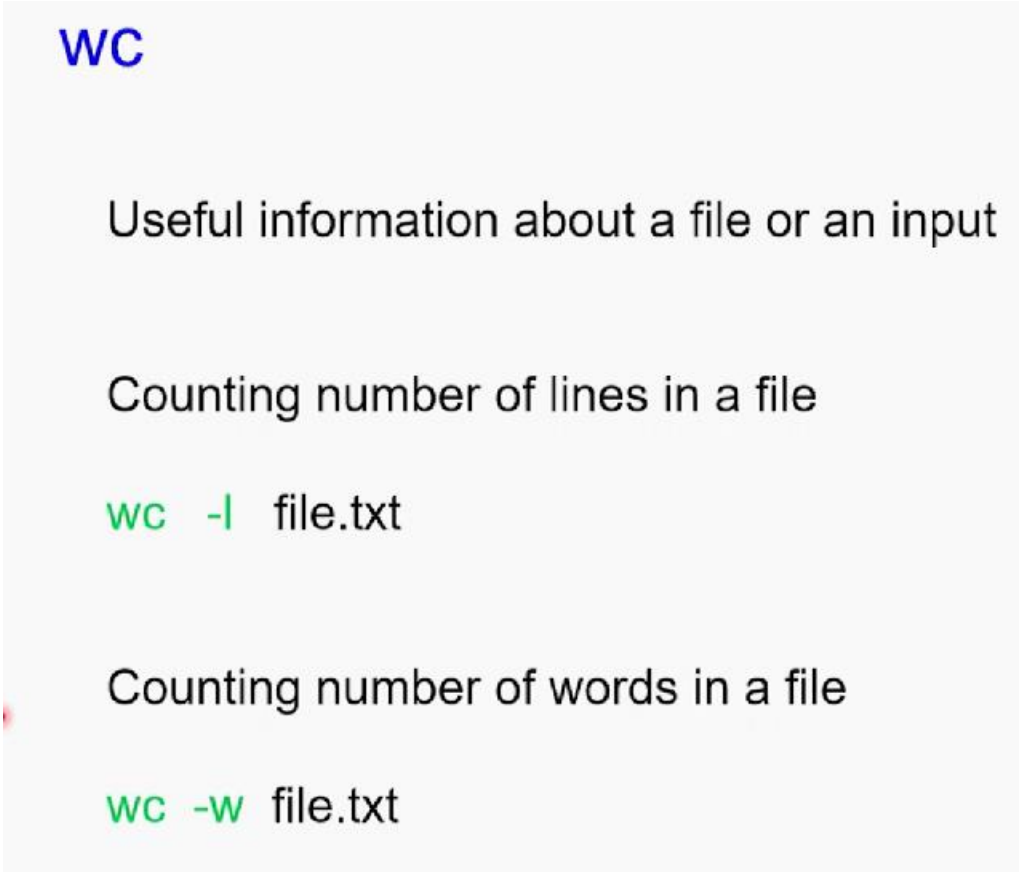
So, we can create this directory, and we can say, Do not extract here; extract in that directory, ok? So, let us do that first, okay? So, how do you create a directory? mkdir right. So, we discussed this in the last class, and we say we create this test 2 inside the test 1 directory, ok? So, if we say that you see this test 2 directory has been created here, right here, test 2 is there.

```
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ mkdir Test2
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ls
Compress.tar.gz                  D12_17539_ATCTCA_read1_part.fastq.gz  Test2
D12_17539_ATCTCA_read1_part.fastq  D12_17539_ATCTCA_read2_part.fastq     compress.zip
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ tar -zxvf Compress.tar.gz -C Test2/
D12_17539_ATCTCA_read1_part.fastq
D12_17539_ATCTCA_read2_part.fastq
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ls Test2/
D12_17539_ATCTCA_read1_part.fastq  D12_17539_ATCTCA_read2_part.fastq
rdhar@LAPTOP-3K4C9VBI:/mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$
```

So, now let us unzip or decompress this, right? tar minus zxvf compress dot tar dot gz right How do you specify the directory minus C Test2? You remember we have just seen OK, and again it will show this has extracted these 2 files, and inside this Test2 directory it should be there. Let us look at the files to see if they are present in this directory. So, the command we use is ls Test2, and

you see these 2 files are there, alright? So, I think this kind of gives you an idea of how to compress or decompress files, especially the decompression, which will be very important for you when you download or extract these tools, and before you use them, you have to do this step. So, let us now go back to the presentation, and we will discuss a few more commands that could be useful in many cases when you are analyzing NGS data.

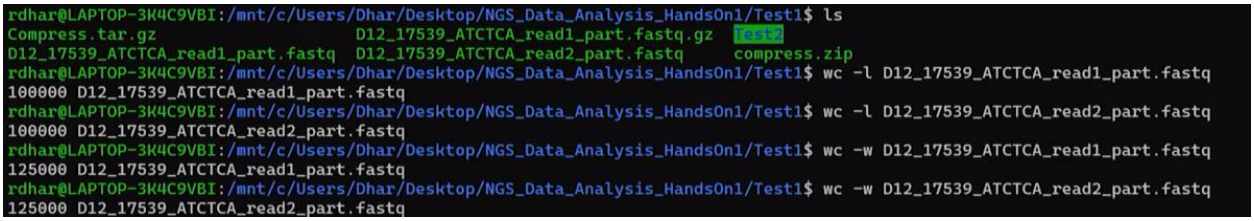
So, the one command that we use is called wc command ok. So, this gives you a lot of useful information about a file on an input. So, this is something that we can use for a variety of applications, but I will talk about only two applications in this case that will be very useful for your problem. So, one thing you can do without actually opening a file is count how many lines are in that file. So, this wc command will tell you how many lines are in that file, ok? So, something called wc minus l ok, the option is minus l and the file name.



So, here the file name is file dot txt. It could be any file that you want to check, or you can simply count the number of words in a file. So, here is this wc minus w file dot.txt, ok. So, let us try this right in the command line and see how what you get from this output is ok. So, we clear the

terminal right so that you can see well, and again, we use the command ls right, and you can see we have these FASTQ files right.

So, you will try these commands wc minus l on these fastq files, ok? So, let us try this wc minus l D12 file name, right? This is a big file name, dot fastq, and we will run it. So, what you see is the number of lines in that file, which in this case is 100,000, and the file name is OK. Now we can do the same for the other files, D12 and D2, and see how many lines there are. So, this makes sense because if you look at these file names, they are read 1, read 2 right they come from the same data; they are paired in data right and contain information about two mates, mate 1 and mate 2. So, they should contain exactly the same number of reads and exactly the same number of lines, ok?



So, that is why we see these 100,000 lines in both of them. Now let us try the other command, which is the wc minus w right, and then the file name. The number of words is given as 125,000 here, and 42 is the same. This is again expected. Now one of the common things you might be thinking about is: how am I writing this big file name so quickly? So, there is a trick in the terminal, which I am going to tell you right now. So, let us we want to let us say we want to run this again wc minus l ok and if you want if you have a big file name right if you just write tab if you paste up tab button it will show you what are the files that are there inside that directory ok. Now, of course, try this only when you are sure that you have only a few files in that directory. If you have 100,000 files, you do not want to try this because it will take a long time to show you all the files.

Now if you want to see right which files start with these letters D12, ok again. So, you type D12, and then you use the tab immediately, and it will give you these extensions automatically. You see, it will find these matches, and it will automatically take this OK. Now it will not go further because it finds two options. Right now, it has this read 1 read 1 underscore part, and the other one is read 2 underscore part. So, now you have to specify which one you want to take. If you want to know what the options are, you simply type the tab twice right, and it will tell you, "Okay, you have these three files, the first part of which matches right with this expression.Then you use, let's

say, "OK," and then press the tab again, and it will take the rest of the file name.

So, this is how we can type really long file names really quickly, ok? You do not have to always type through the letters because that is going to take a lot of time. You can simply type a part of this file name and use tab OK. So, this is a very useful way of writing very big file names, ok? So, you have understood how to write wc minus l and wc minus w. Now let us go to the other important expression that is actually very useful, as I have mentioned earlier.

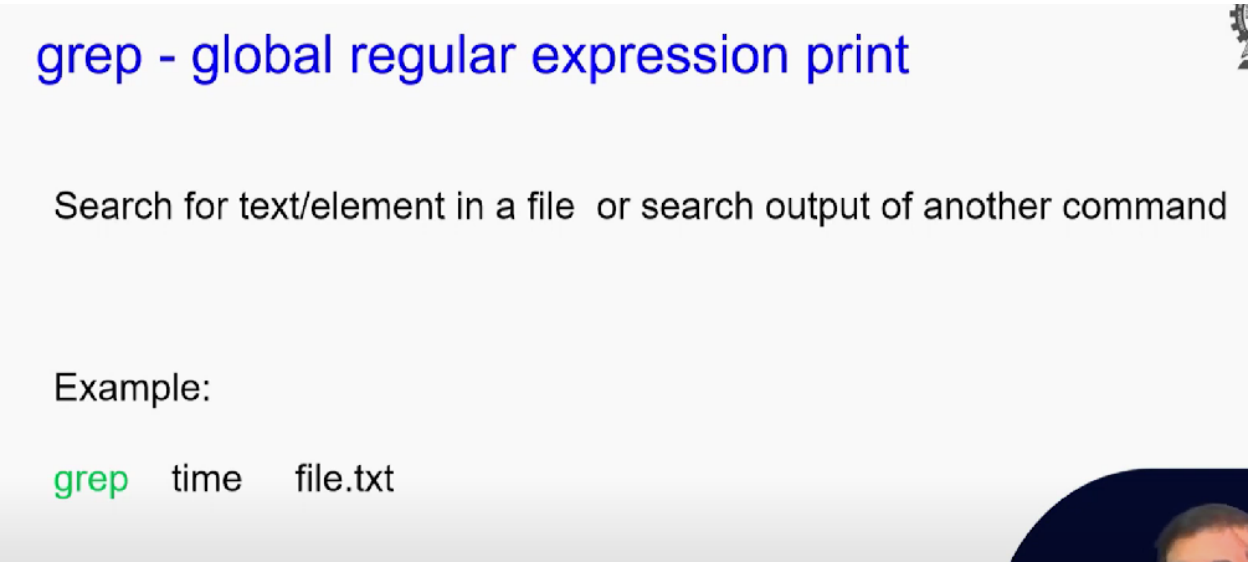
So, this is called the grep, okay? So, the full form is the global regular expression print OK. So, what is this global regular expression print? So, as I said, if, for example, you want to find a file name, let us say a read name in your file, or it could be. So, if you are not talking about NGS data for any file name, you want to find a specific expression, a specific name, or a specific data point that has a unique name. So, for doing that, we use this command, creep, ok? So, it can search this whole file very efficiently, and it can give you the instances where this has found this expression.

So, this command is used, as I mentioned, to search for text or elements in a file, or you can also search the output of another command. This is something we will not discuss, but you can, of course, look it up. So, if you are utilizing another command in this Linux command line or terminal, you can actually pass that output of that command through grep right to search for specific outputs or specific patterns. So, this is something that can also be done right.

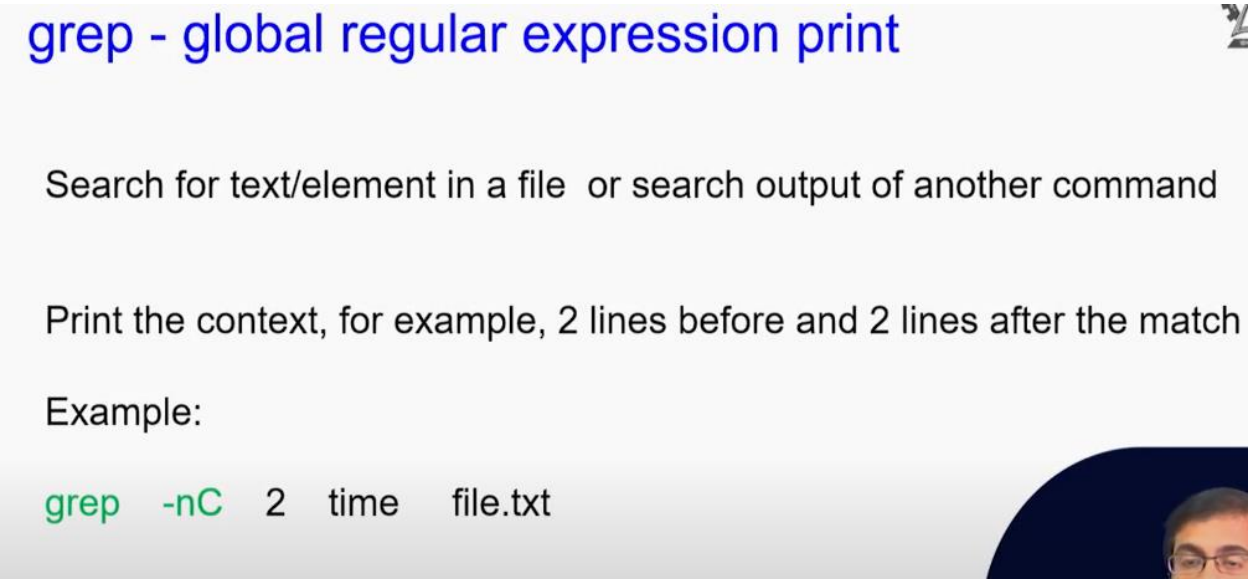
So, this is a very useful tool; it has a lot of ways of applying it. So, you will probably learn as you go along, like if you learn more about Linux. We will use this for a very simple kind of task, ok? We want to find the specific name inside a file, okay? So, for example, we want to find a read name inside a file, or we want to find, for example, the mutation data once we have done that NGS data analysis. We want to look for certain chromosomes, right? Maybe there are mutations in certain chromosomes that we want to look at, let us say chromosome 5 in the human genome, because that might be useful for certain disease identification, classification, etcetera.

So, this is a very simple task, and grape could be very useful for many applications. So, this is how the expression goes, right? So, grep is the command you have this time, which means this is the

text that we are searching for and then we have this file, dot.txt. This is the file where we want to look this time. This is the expression that you want to look for in that file, ok?



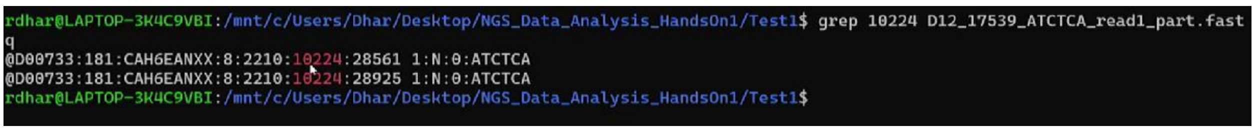
Now let us look at this right. We will see this in a moment in the command line. There is another option that you can also take, right? For example, you can do what you want, like, you just want the occurrence you want to get and also the context of this file name. So, context means maybe just show us the first two lines before the occurrence and the two lines after the occurrence. So, Grep can also do that right.



So, this is where we use this right. So, grep, the option is minus n capital C, right? So, note these options, whether they are in capital or small letters. This is very important right? These are actually critical. You cannot just interchange small and capital, ok? So, this is something you should note

for all commands, ok? So, if you use this minus nC 2, right? So, this means print the context two lines before and two lines after the match, ok? So, you are searching for this expression time in this file, and wherever there is a match, the command will also print two lines before the match and two lines after the match, which kind of gives you context.

This is useful for some applications where you probably want the context as well. Maybe there are multiple matches, but you are probably looking for just one or two places or two contexts. So, this is something that is very useful. So, let us try this out right on the command line. So, what we will search for here, ok, we can search for certain read names, etcetera, right?

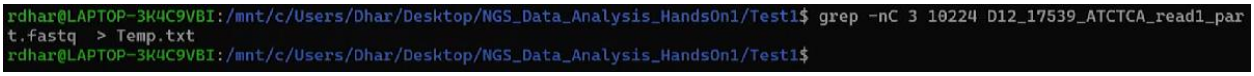


So, let us search for certain names, okay? Maybe we can search for a certain number. So, let us try to see if we can find this right. This number probably will occur in the header line. This is a FASTQ file, so this number might occur in the header line. So, let us see if we can find a match somewhere, okay? So, this is the expression. Write grep 10224 D12, read the file name, and we will search, and it will return two results. It has found this number in two places. So, you can search for anything you want using this command, and it also highlights right here that it has found this in this red color, ok? Now let us try the other one. We want some context, not just the results but also the context, ok?

So, this is the format again. We use this option minus n capital C and 2 right. We want two lines. If you want three lines, let us. In case we want three lines, we say 3 the expression that we are

looking for and the file name. And this is what you get. Maybe it is slightly cumbersome; it is difficult to read at this point right? So, it gives you the context, and the line number is okay. So, where is it actually finding this in the file? So, on the right, in green, you see the line number right where it is finding this and some of the data.

So, here it has found this 10224, and it is giving this three lines before this and three lines after this. And similarly, for the second match, it will also give you context. Now you can also take this out and store it in another file, or maybe in the terminal. If you are seeing this output, it becomes very cumbersome. So, you can say store this in a temporary file, Temp dot txt, ok?



And you can simply channel this output using the present sign. So, it will go into this temp dot.txt, ok? And let us do that and see if it will not print anything here in that in the terminal; it will write everything in the Temp dot txt. Now how do you actually get this Temp dot txt? We have used certain commands, right? We have remembered, right? We have discussed, how do you get this in the terminal from a file? You can use cat, right? You can say cat temp dot txt, or you can also use less right temp dot txt. You discussed this in the last class, ok?

And as you can see, this output file now contains the output of that command line. So, this kind of helps in storing whatever you have done. So, just to summarize, this is the reference that we have

used and to summarize, we have discussed shell commands that allow us to navigate through unique systems as well as data compression and decompression commands. And these commands are really useful for many tools and software that will be used for NGS data analysis. Thank you. Thank you.