

Next Generation Sequencing Technologies: Data Analysis and Applications

Hands-on 1 – Data exploration and QC

Dr. Riddhiman Dhar, Department of Biotechnology

Indian Institute of Technology Kharagpur

Good day, everyone. Welcome to the course on Next Generation Sequencing Technologies, Data Analysis, and Applications. In this lecture, we will be discussing basic technologies, data analysis, and applications. So far, we have set up the system for hands-on and learned basic Linux commands. So, we will utilize those commands today to look at the data, and then we will do the quality control of the test data. So, let us start.

So, these are the two concepts that we will be talking about today. So, we will talk about the data exploration. We will utilize some of the command-line tools to look at some of the NGS data, and then we will do quality control on a fast queue data set. So, these are the steps. So, the first step is the data download.

Now, you might be doing your own experiment. So, you will get this data from the sequencing center or the company where you are doing your sequencing. So, you will need to simply put that data into your system, or if you are working on a server, you need to get the data to the server system, ok? If you do not have your own data, do not worry. So, you can actually download some of the freely available data, and you can test these pipelines and these commands yourself, ok?

So, I will give you some links here where you can actually download these data sets, ok? These are part of some studies that have been deposited in these databases. And once we have the data set, we can look at the data. We can look at how the data looks and some of the aspects, and we will also learn a little bit about the VI editor and how to use the VI editor to navigate the data. It is not just for navigating fastq data; you can utilize it for navigating any files or data sets. So, there are certain commands with VI that make it really convenient to use, especially if you are dealing with a really big data set.

So, this would be the first step. We will look at some of the data and how you actually download it. I will show you in a moment, and we will also explore these data sets with the VI editor. We will learn a little bit about the commands that you can use with VI. And then we will actually do the quality control with fast queue data.

So, for that, we need the fast queue-seeing tool that we have already discussed in the theory class, right? So, we will download this fastQC tool, set it up, and then do the data quality control. We will actually run our data through this fastQC pipeline, and finally, we will do the quality evaluation, which means we will look at the output. As we have discussed, we will generate certain statistics for you, and we need to make sense of those statistics, so we will have to make decisions ourselves. What do we want to do before we actually go for downstream analysis? Do you need any pre-processing? Do you need any trimming? Those decisions would have to be taken by us, ok?

So, let us start. Let us look at some of the links where you can find these data sets so that you can download them, or if you have your own data, you can test all these commands with that data set. So, let us go to the links, ok? So, here are the links that I actually shared with you, ok? And this is the first one: an NCBI-GEO, Gene Expression Omnibus database, where you can actually look at hundreds of data sets.

So, there will be different types of data sets, and you will see this kind of table when you open this site, ok? Now, just to make sense of it a little bit, right? So, here you see something called an accession number. So, this is a unique number that is given to each study. You will have a title for that data set, right? What is this data set about? What kind of study is it?

Then you have something called a series type, right? This is whether this is expression profiling, whether this is methylation profiling, and what kind of experiment it is, right? So, again, I will specifically mention whether this is expression profiling. So, as you can see, expression profiling by high-throughput sequencing means you get transcriptomic data generated through NGS platforms. Then you have this column where you get the

organism's name, ok?

So, here is a human data set, but you can have other types of data sets. For example, here you see mice, *Vibrio cholerae*, etcetera, etcetera. Then you have a column with samples, right? And then, of course, you have some contact information released there when the data was submitted, etcetera. You can simply click on this entry, right, and you will see a lot more details about this data set, ok?

So, maybe you can zoom in a bit, and you can see. So, this is single-cell RNA sequencing of ILCs and T cells, right, and a lot of these details from this disease, etcetera. What is important? You see, this is expression profiling by high-throughput sequencing, and then you have a summary. Now, these summary files are sometimes very important because they give you an idea of the study design, right, and how this study was conducted. And before you do any analysis, you probably want to frame your questions about what kind of analysis you want to do, and then you utilize this data set.

If you go down, you will see some of these ideas about these platforms, right? Here, this platform is Illumina HiSeq4000, ok? So, we have talked about Illumina platforms and how they work, and then you have the sample names, right? And finally, you have the data here, down here, ok? So, we will look at this kind of data. This is transcriptomic data when we actually do the transcriptomic analysis when you go into the RNA-Seq data analysis.

You have other data sites, right, where you have this, for example, the European nucleotide archive. You can again look at some of the data sets that are here, or you can also look at something called biostudies at Express, where again you have similar kinds of data. So, on the left, you can actually choose which kind of data you are looking for, whether you are looking for methylation profiling by negative sequencing or maybe something else. So, this is something that you can get again. All right, so once you have this data, let us now go to the terminal, where we can look at these data sets.

So, here I will utilize one data set that I have already downloaded into the system, and we

will do all the work on that data set, ok? So, let us have a look. So, you probably remember these commands—the basic Linux commands that we have talked about, right? So, this is `ls`. It gives you the list of files that are present there, or you can also use `ls -l`.

It gives you a lot more information, right? On the left side, as you have seen, we have already discussed the permissions of these files, and then we have the file names and the size of these files. You can see this. You can probably compare the size of these files. What we are interested in are these two files, ok?

You have this `read1part.fastq`, and then you have `read2part.fastq`, right? These are the FASTQ files. Now, why do we have these names, right? So, `read1`, and `read2` means this is paired in data, right?

Immediately, it is clear. The rest of the name is the same, right? So we have this `read 1`, `read 2`. So, this is paired with data. You have this information for two different reads, right? So, from sequencing from both directions, ok.

And then you have this `part`. This name `part` means this is a part of the original file, ok? So, as we have discussed, these fastq files, the data that is generated from the Illumina platform, are actually huge in size—ok, really big because you have millions of reads in there. Now, if you want to do all these analyses with that kind of dataset, you will not be able to complete these hands-on activities within a reasonable period of time, because all the steps will take a long time to complete. So, that is why I have taken only a part of the dataset, right, so that we can do this analysis. I can show you all the steps in a reasonable amount of time, and in the pipeline, the process remains exactly the same, right? Whether you are dealing with 100,000 reads or millions of reads, the process does not change, ok?

It will just take more time for the whole process to complete, ok? So, we will work with these `part` datasets, ok, `read1 part.fastq` and `read2 part.fastq`, ok. So, the first thing is that we will actually use the VI editor to look at this data and we will discuss certain commands that are very useful for navigating through these files.

you can see we have about 10,000, sorry, 100,000 lines here. You can see this right here:
100,000 L means 100,000 lines.

Each read is taking four lines, which means we have 25,000 reads, ok? So, we have separated out 25,000 reads, and we will work with these 25,000 reads for all the analysis. So, one of the things you probably notice is that the sequence does not seem to be in one line, ok? So, it is because of the wrapping, right?

So, text wrapping by the editor is okay. So, if you want to remove this wrapping, there is a command in vi that you can use. Now, there are two things that you can do with Vi, ok. So, Vi can be run in read mode, which we are in right now. So, whenever you open a file in vi, it will be in this read mode, ok? So, read mode means you cannot write anything, ok, and this read mode exists because we want to prevent any accidental writing on important files, ok.

So, this is the read mode, but you can enter the write mode when you are sure you want to modify certain parts, add some data information, etcetera. Now, to enter this write mode, you have to press I. The moment you press I, what you get is this insert, ok? You will see this here, right? All these details are here, right at the bottom.

So, you see, this insert means you are in write mode. If you want to come out of this write mode, you have to press esc ok, and that insert command is gone. So, you can go into this write mode, and you can start looking at this file. Now, in the read mode, you can run certain commands in Vi. So, the one command that I mentioned is this: we do not want any wrapping around, right?

We want this read sequence in one line. This is the format of FASTQ. So, how do you move this wrapping? So, this is the command. You can follow this here, right? I am writing these commands here, and what I see is a colon, right, and then you have no wrap.

We do not want wrapping. So, we just say no wrap, and you can see the format has changed,

ok? So, the sequence data now is in one line, then you have the spacing, and you have the ASCII encoded quality score, ok? So, this is the actual format of FASTQ, ok? And now, if you want to go to the end of the line, how do you actually go? It is simply a shift.

So, if you press simply dollar, ok. The moment you press dollar, you go to the end of the line, ok and you can also come back to the beginning of the line by pressing this caret side, ok. So, you probably know what this caret side is. So, this caret side will bring you back to the start of the line. So, these are really useful functions, right? You do not need to traverse through all these elements to get to the end of the line.

You can very quickly go back and forth, ok? Now, one of the questions you might have is: can you go up and down very easily, right? And the answer is yes, we can. So, if you want to go to the end of the line, you can use these arrow keys, of course, but that will be cumbersome, right? You have to traverse through the full file. There is actually a faster way to get to the end of the line. So, it is actually a colon dollar, ok?

So, here we are, right? You can see we are at the end of the line. Now, on the right side here, if you look right at the bottom, you can see these numbers and this letter BOT bottom, ok? So, this means we have reached the end of the file, ok? And of these two numbers here, the first number is 100,000.

This is the number line number, ok? And this is this number 1; after this comma is the column number, ok? So, you will see if we move to the right. You can follow the cursor here. If you are moving to the right, this number changes, right? We are moving into these columns, right?

So, this is actually counting the column number. Can you move to the top from the bottom? Yes, we can. We can simply say colon 1. We want to go to the first line. So, we can mention the line number, and we will reach the first line.

So, this works for all the lines. If you want to go to the 100th line, you can say 100, ok, and

you go to the 100th line, ok. So, these are really useful, right? You can see you can very easily traverse through these files without actually going through the whole file, right? Now, there are also other functions. You can search through these files very easily, like with `grep` in the command line.

We have seen how `grep` works, which is in the command line, but inside Vi you can also search, ok? And this is how it works, ok? You can simply press slash, and you can give some text or numbers that we are looking for, and you will see right away that the moment I have typed, this part is highlighted, ok? It means it has found this number here, ok, 10224. You can press enter, and it will then if you want to search it again, just press it again, and you will find another instance of this.

And at the same time, it will tell you, right, where it is finding these terms or these numbers, ok? This is the line where it is found, and this is the column, and it also gives you an idea of where in the file you are. So, here is 31 percent, right? We have traversed 31 percent of the file, right?

So, somewhere in the middle here, we are finding these terms, ok? So that is a great thing, right? We can kind of search; we can traverse through these files very, very easily, ok? So, let us go back to the first line, ok? And one of the things that you might be wondering is that if we can write anything on this file, ok.

So, to do that, we have to go into write mode, ok? Now, before we go into the write mode, one of the things I will say is that if you are just interested in the read mode, for example, with first queue files, we do not want to write anything. So, we just want to come out of that file. So, how do you come out? So, to come out of the VI editor, we type colon q, ok? So, it will simply take you out of that file.

It will close that file, and you will come out of that file, ok? So, this is very convenient, right? In the terminal you can go into a certain file, open a certain file, work on that file, or come out of that file. So, how do you write here? So, this is something we will discuss

now,

ok?

So, let us remove this wrapping, right? And let us enter write mode, ok? So, here we are; we have entered write mode now, ok? And let us say we write something like x y x y or something like that, ok? We are just writing anything that we want, ok? And so once we are done, we can come out of that writing mode by pressing escape.

Now it turns out there are a lot of other functions that are very helpful for your work, ok? So, for example, if we have written something and you want to copy it multiple times, So, there is a shortcut in VI, right? In the read mode, you can simply press 2 y 2 twice, and it will copy the line where your cursor is.

So, we are in this line. So, if we press y twice, it will copy that line. And if you want to paste that line somewhere in the file again, right, we go to that line where we want to press it and paste it, right. And we press p, and that line will be pasted, ok? So, this is very convenient in that we can copy this line very easily with these commands, ok?

So, this command is yy. There is, so this is kind of a copy-and-paste command, right? This is equivalent to copying and pasting in Windows. There is also cut paste. So, that works by saying d d, right, small d's, right, twice.

If you press d, this line will be cut. And if you let us say you want to paste it here somewhere, you can again say p. You can press p, and then it will be pasted here, ok? Now, this is copying one line or cutting one line, right? What about copying 40 lines, ok or 50 lines, all right? So, how do you do that? So, the idea is very simple, right?

So, if you want to copy, let us say 40 lines; you simply say 40 y y. And you will see this here; it says 40 lines, which means 40 lines copied, ok? And you want to press and paste those 40 lines here; maybe after this line you simply say p, and we will paste those 40 lines here, ok? So, this is 40 y y, and then p, ok, p for pasting, and y is for copy. Now, you can do the same thing for cut and paste.

You can say 40 d d; it will cut 40 lines, and it will paste 40 lines if you press p. If you do not press p, it will simply delete those 40 lines. Now, if you say we do not want to change FASTQ files, right? So, we want to undo all the changes that we have made, ok? So, how do you do that? You simply say, Simply press u, right?

And you can press it multiple times, right? It will go back to the oldest format, right? As long as you have not exited the vi, all those changes will be removed, right? All those changes that you have made after opening the file will be removed if you press u, ok? And then we can go out, saying colon q, ok. Now, if you want to, if you are writing anything and you do not want to save it, let us say we again insert something by mistake, but we do not want to press u so many times.

We want to just exit, but we do not want to save the changes that we have made in this file. So, how do we do that? We say colon q and then an exclamation mark, ok. So, the exclamation mark means quit without saving, ok? Whatever I have written by mistake, we do not want to save this.

So, we just want to quit without saving, ok? So, we can go back and you will see that these changes are gone, right? This is h, s, h; whatever I had written earlier, that is gone. Now, at times you want to probably make some changes, and you want to save those changes, right? So, again, we go into insert mode, and we want to write something, ok, right?

I am writing something again, but this time I want to save it, ok. So, the command to save and to exit is colon wq, ok, and here we are, out of that file, ok. And if we go back again, we will see this line is there, ok, because we have exited by saving, right? So, this line will be there, alright? So, these are some of the basic commands that we can use. Of course, there are many more commands and many more functionalities, and that is why Vi is really powerful, right?

We can do a lot of things in the terminal just by using Vi and opening the files, and there

are commands within Vi that work very well and can help us navigate these files very easily, ok? We do not have to always traverse through the files or through the lines. We can do a lot of things very easily using simple commands inside Vi. So, I will delete whatever I have written because, at the end, we do not want to change the fastq file, and we will go to the terminal. Now, it is saying there was a change to if you do not want to save that change at this exclamation mark, right? You see this command here.

So, at times, you get this kind of feedback, right? If you are trying to do something that you do not, it is kind of like giving your warning, right, so that you do not make some mistake, right? So, we do not want to save this change.

We want to save this change, right? We have removed this. So, you have to say colon wq, right? We have removed whatever we had written, ok? And now here we are, ok? So, this kind of summarizes, right, what you can do with Vi, what you can do with how you can navigate these files and modify files, add contents, write things, delete things, copy things, right, inside the file, ok? So that gives you an overview.

So, now the next agenda item is actually to look at the quality of these FASTQ files. So, to do that, we have these steps that we have talked about. The first is setting up the FASTQC. We will download and set up FASTQC, and then we will run FASTQC, right? We will generate this data, which will be quality data, and it will give us some idea about the quality of these datasets, ok?

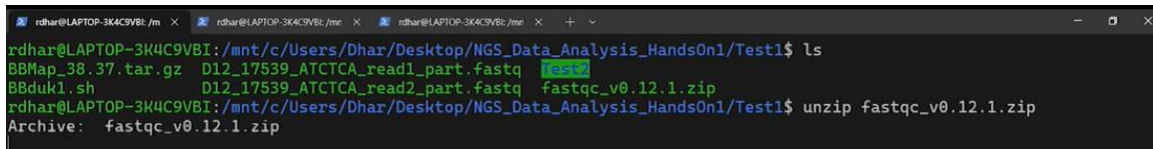
So, let us start with that now. So, I will just clear the terminal, and we will start with the FASTQC download, ok? Now, the installation of FASTQC is actually very, very simple. If you can just type FASTQC in Google, the first page that you will get is this tool, ok? So, you can simply click on that tool, right, and this is the page where you will be, ok?

So, here you can download this, ok? Here is the link to download now, and you have all the instructions. You have readme, you have installation, and you have everything in here, ok? So, here, what you have to do is just look at these two files, ok?

One is for Windows and Linux systems. This is a zip file. The other is for the Mac system, ok? So, if you are working with Windows or Linux, download this file. If you are working with Mac, you can download this file, ok? So, I have actually downloaded this because it might take a bit of time for me to do so. So, I have downloaded it already, and I will show you how we can install it and run it in the terminal.

So, let us do that. So, here it is, right? Here is the FASTQC, ok? Here is the FASTQC, right, and we can extract this in here, right, inside this, ok. So, here you have a copy of this FASTQC. Maybe I will zoom in a bit so that it is probably easier for you. Okay, all right. So, here is the FASTQC zip file that you have downloaded, and we can simply extract this, right, and we can also try the command line, right, to extract this.

Since you know the command line, try to utilize that, ok? So, this is simply unzip and let us see, ok? So, this is done; this is okay. So, this is done. You can clear and you can simply say ls, ok, and you will see this folder FASTQC that has been created, ok. So, in this FASTQC folder, right, inside this you will have that program, ok?



```
rdhar@LAPTOP-3K4C9VBI: /mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ls
BBMap_38.37.tar.gz  D12_17539_ATCTCA_read1_part.fastq  test2
BBduk1.sh          D12_17539_ATCTCA_read2_part.fastq  fastqc_v0.12.1.zip
rdhar@LAPTOP-3K4C9VBI: /mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ unzip fastqc_v0.12.1.zip
Archive:  fastqc_v0.12.1.zip
```

So, how do you go inside that folder? The command is cd, right? So, we move into that folder, ok, and here you have a lot of these things, a lot of files you can see, ok, but do not worry, right, we do not have to worry about any of these things too much. The only file that we have to work with is this one, FASTQC. This is the file that will execute, and this will generate the quality scores in the system, ok? So, let us look at this permission of this file, ok, because if we do not have the execution permission, we will not be able to write this, ok, because sometimes when you download these files and extract, we might not have this execution permission, right?

That is the only permission we need. We do not need read or write permission on that tool,

ok? So, let us look at this, right? So, how do you check permission? This is the command
ls minus l here, right?

We are inside the folder. So, this will give you permission, ok? So, this looks great, right?
So, if you look at this FASTQC file, we have read, write, and execution permission, right?
So, we can execute this code. We do not have to worry about anything here, ok? So, let us
now execute, ok, and see how we generate the quality, ok.

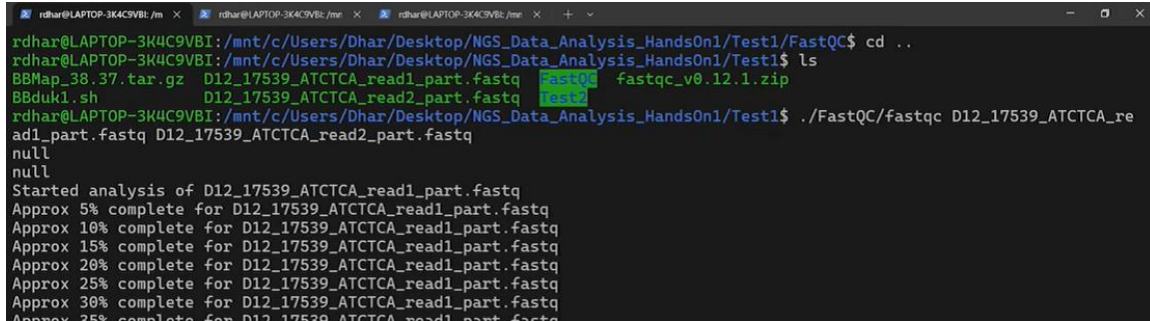
So, there are two ways you can run FASTQC on the Windows system, in particular. So, in
the Windows system, you can run it by just clicking on that FASTQC inside that folder,
right? If you have Java installed, it will simply open up a window, and you can load a
FASTQC file and generate all the quality statistics. There is a better way if you are running,
especially WSL inside Windows; you can run FASTQC through the command line itself,
ok? And on Mac, you can also run through the command line. So, what is the advantage of
running through the command line? Because if you have a lot of data and you want to do
quality checks on all of it, it is better to run through the command line because you can run
using a single command.

If you are running through that program window, you will have to load individual files,
right? You can do only one file at a time, and you will have to do this repeatedly. If you
have, say, 10 files, you have to do this 10 times. So, we will prefer using the command
line, ok? So, let us utilize this command-line tool and then do the quality check, ok? So,
how do you call this program FASTQC inside this folder?

So, to run that, I will just come out because our data files are here in this folder, right? So,
we need to give these data files to that program, ok? So, I will now call this program, right,
because this is inside this folder. So, I will have to go inside that folder to call this program,
ok?

And you notice this is a dot, right? I have given this dot slash, which means we are
executing this program, right? We are calling this program in there, ok? So, this is how we

will call this program, right? So, and then we would have to give the names of the files on which we want to run this quality control, ok? So, here are the files we have, and you can specify multiple files, ok, one after another.



```
rdhar@LAPTOP-3K4C9VBI: /mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1/FastQC$ cd ..
rdhar@LAPTOP-3K4C9VBI: /mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ls
BBDuk1.sh          D12_17539_ATCTCA_read2_part.fastq  fastqc
BBMap_38.37.tar.gz D12_17539_ATCTCA_read1_part.fastq  fastqc_v0.12.1.zip
rdhar@LAPTOP-3K4C9VBI: /mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ./FastQC/fastqc D12_17539_ATCTCA_re
ad1_part.fastq D12_17539_ATCTCA_read2_part.fastq
null
null
Started analysis of D12_17539_ATCTCA_read1_part.fastq
Approx 5% complete for D12_17539_ATCTCA_read1_part.fastq
Approx 10% complete for D12_17539_ATCTCA_read1_part.fastq
Approx 15% complete for D12_17539_ATCTCA_read1_part.fastq
Approx 20% complete for D12_17539_ATCTCA_read1_part.fastq
Approx 25% complete for D12_17539_ATCTCA_read1_part.fastq
Approx 30% complete for D12_17539_ATCTCA_read1_part.fastq
Approx 35% complete for D12_17539_ATCTCA_read1_part.fastq
```

So, we have two files. We will run them here. We will mention this read-one-part dot FASTQ, followed by read-two-part FASTQ. So, as you will see, FASTQC will run this quality control one after another, ok? So, if you have ten files, it will run this for ten files consecutively, ok? So, we are running this, and it gives you an update, right, on what percentage of the run is complete, etcetera, for each of these.

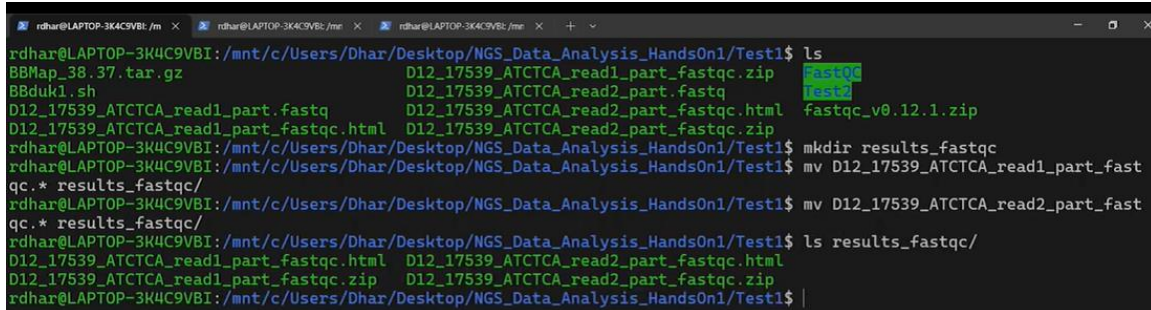
So, once it is done, right, it will give you this, and it will be back here, right? So, this is done now, ok, and we can now go and check the quality score, ok, ok. So, let us go inside that. So, you will see that inside this folder, these four files will be created.

We have run two FASTQ files. So, for each, there will be two files generated by FASTQC, ok? So, you have this read-one-part FASTQC, right? This is HTML, and then you have the zip file, ok? This will be clear if we actually do this in the terminal, ok?

So, let us do this in the terminal, and you will probably see this clearly, ok? So, you can see this FASTQC.html, ok? This has been created, and for read one, this is another file that has been created, right, FASTQC.zip. So, all the results are inside this, inside these two files, ok, and then you have similarly for read two, you have these two files created, ok.

You have read two parts of FASTQC.html and two parts of FASTQC.zip, ok? So, what we will do is create a folder saying results_fastqc, right? So, we can move all the FASTQC

results inside that folder, and we can then check the results and make decisions, ok? To create this folder or directory, use mkdir in Linux, and we will move these FASTQC files. So, we want to move these FASTQC files, right, and the command is mv, right, move, ok.



```
rdhar@LAPTOP-3K4C9VBI: /mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ls
BBDuk1.sh
D12_17539_ATCTCA_read1_part_fastq
D12_17539_ATCTCA_read1_part_fastqc.html
D12_17539_ATCTCA_read2_part_fastq
D12_17539_ATCTCA_read2_part_fastqc.html
D12_17539_ATCTCA_read2_part_fastqc.zip
fastqc_v0.12.1.zip
rdhar@LAPTOP-3K4C9VBI: /mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ mkdir results_fastqc
rdhar@LAPTOP-3K4C9VBI: /mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ mv D12_17539_ATCTCA_read1_part_fastqc.* results_fastqc/
rdhar@LAPTOP-3K4C9VBI: /mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ mv D12_17539_ATCTCA_read2_part_fastqc.* results_fastqc/
rdhar@LAPTOP-3K4C9VBI: /mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ ls results_fastqc/
D12_17539_ATCTCA_read1_part_fastqc.html
D12_17539_ATCTCA_read2_part_fastqc.html
D12_17539_ATCTCA_read1_part_fastqc.zip
D12_17539_ATCTCA_read2_part_fastqc.zip
rdhar@LAPTOP-3K4C9VBI: /mnt/c/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn1/Test1$ |
```

So, we have this part FASTQC star and then the destination directory, right, where we want to move, ok? Hopefully, you remember the basic commands that we discussed, ok? And similarly, we have done this for read one; we will do this for read two as well, ok? And you might be wondering, right, why I am giving this file the name dot star, ok? So, star is a wild card, which means it will match anything it finds, ok?

So, at once, it will move this HTML as well as zip inside this folder, ok? So, this is moving it, and now we can check the results, right? You can see all the FASTQC results are inside this folder. We can move now inside this folder, and we can unzip, right? We can simply unzip the zip folder so that we can check the results, ok?

And we do this for both of them, right, for read one and read two, and we have now all the results created. You can see that these two folders have been created. These are unzipped ones, and inside this you will have the results, ok? Now, we can go inside this, do the quality evaluation, and make our decisions, ok? So, in the next class, we will actually look at this data, right? We will look at this folder; we will look at all the results that have been generated, and then we will make decisions and do the preprocessing. So, with that, I will stop.