QAA

Four input files for two samples. No index files because this data has been demultiplexed. so R1 R2 for each sample.

Mahmoud | 31_4F_fox_S22_L008 10_2G_both_S8_L008

What QC programs should tell us:

- per base qual score
- per base N
- GC content (can check for PCR bias "less GC and more AT")
- length distribution
- adapter content
- over represented sequences
- number of sequences
- encoding

In FastQC statistics:
quality scores for R2 read is always less than R1 because it is sequences last

per tile sequence quality can tell us about bubbles interfering with the quality of sequencing.

We don't care about sequence **duplication** levels in **RNA seq experiments**, because the point of RNA seq is to see the levels of expression, so if something is over expressed, we want to see that.

We see adapters in our sequences if our read is short. We want to trim them out if they are present in our data.

Once we run the software from the terminal, the individual plots will be placed in the zipped files.

Part 1:

Creating Conda environment:

```
srun -A bgmp -p bgmp --time=0-3 --pty bash
cd /projects/bgmp/malm/bioinfo/Bi623
conda create -n QAA
conda activate QAA
```

To download fastqc tool

- went to their website and copied the tool link for linux/windows "https://www.bioinformatics.babraham.ac.uk/projects/download.html#fastqc"
- In the terminal I ran:

```
wget
https://www.bioinformatics.babraham.ac.uk/projects/download.html#
fastqc
unzip fastqc_v0.12.1.zip
chmod 755 FastQC
FastQC/fastqc --version
> FastQC v0.12.1
```

*To run fastqc in this setup, we would always have to give it's path. To be able to run it from different directories without giving its path every time; we can add its path to our bashprofile
If the path is wrong here it will mess up a lotttt of things. BE CAREFUL

Editing .bashrc:

```
cd
cp .bashrc .bashrc.copy
vi .bashrc
export PATH=$PATH:/projects/bgmp/malm/bioinfo/Bi623/FastQC
```

now running `fastqc --version` from anywhere should give us the right output as above

To find the manual on how to run fastqc look at the INSTALL.txt file which is one of the files included when downloading the tool

```
cat INSTALL.txt
```

> To run non-interactively you simply have to specify a list of files to process on the commandline

`fastqc somefile.txt someotherfile.txt`

> There are a few extra options you can specify when running non-interactively. Full
> details of these can be found by running

`fastqc --help`

> If you want to save your reports in a folder other than the folder which contained your original FastQ files then you can specify an alternative location by setting a

`--outdir value:`

my bash script for each file was the following (with the difference of file names) called `fastqc1.sh and fastqc2.sh`

```
#!/bin/bash

#SBATCH --account=bgmp

#SBATCH --partition=bgmp

#SBATCH -c 1

#SBATCH --time=0-3

conda activate QAA


./fastqc
/projects/bgmp/shared/2017_sequencing/demultiplexed/10_2G_both_S8
_L008_R1_001.fastq.gz
/projects/bgmp/shared/2017_sequencing/demultiplexed/10_2G_both_S8
_L008_R2_001.fastq.gz --outdir
/projects/bgmp/malm/bioinfo/Bi623/FastQC/fastqc_output2
```

- *This fastqc run took much longer than the first one since these files are so much bigger

Then to open the html files, I tried firefox but it didn't work. `lynx <filename>` works but does not look good. So i downloaded the files to my computer using

```
scp
ta:/projects/bgmp/malm/bioinfo/Bi623/FastQC/fastqc_output1/31_4F_
fox_S22_L008_R2_001_fastqc.html .
```

And I was in my Desktop directory, then opened the files normally using safari.

*To generate the plots with per base quality scores using my code written for the Demultiplex assignment, I had to make some adjustments.

- Leslie recommended that I don't use sys and argsparse together, so I removed sys and added a file name flag (-f) to my argsparse to give the filename. I copied my `bioinfo.py` and the adjusted code to a new directory here `/projects/bgmp/malm/bioinfo/Bi623/FastQC/mycode-to-make-plots` where the plots will be generated and then compared to the fastqc plots.
- It took around 1 minute to generate the plot for each R1 and R2 of sample 1, white it took around 17 minutes to do the same task for sample 2 that is much larger.

Part 2:

- **To Install cutadapt and trimmomatic tools I used conda:

```
conda install cutadapt
conda install trimmomatic

cutadapt --version
4.9
trimmomatic -version
0.39
```

`

`Run cutadapt -help`
to get directions on how to run this tool

From cutadapt user guide:

3’ adapters

A 3’ adapter is a piece of DNA ligated to the 3’ end of the DNA fragment you are interested in. The sequencer starts the sequencing process at the 5’ end of the fragment and sequences into the adapter if the read is long enough. The read that it outputs will then have a part of the adapter in the end. Or, if the adapter was short and the read length quite long, then the adapter will be somewhere within the read (followed by other bases).

For example, assume your fragment of interest is *MYSEQUENCE* and the adapter is *ADAPTER*. Depending on the read length, you will get reads that look like this:

```
MYSEQUEN
MYSEQUENCEADAP
MYSEQUENCEADAPTER
MYSEQUENCEADAPTERSOMETHINGELSE
```

Use cutadapt’s `-a ADAPTER` option to remove this type of adapter. This will be the result:

```
MYSEQUEN
MYSEQUENCE
MYSEQUENCE
MYSEQUENCE
```

As can be seen, cutadapt correctly deals with partial adapter matches, and also with any trailing sequences after the adapter. Cutadapt deals with 3’ adapters by removing the adapter itself and any sequence that may follow. If the sequence starts with an adapter, like this:

***ADAPTERSOMETHING**
Then the sequence will be empty after trimming. By default, empty reads are kept and will appear in the output.

5’ adapters

Note

Unless your adapter may also occur in a degraded form, you probably want to use an anchored 5’ adapter, described in the next section.

A 5’ adapter is a piece of DNA ligated to the 5’ end of the DNA fragment of interest. The adapter sequence is expected to appear at the start of the read, but may be partially degraded. The sequence may also appear somewhere within the read. In all cases, the adapter itself and the sequence preceding it is removed.

Again, assume your fragment of interest is *MYSEQUENCE* and the adapter is *ADAPTER*. The reads may look like this:

```
ADAPTERMYSEQUENCE
DAPTERMYSEQUENCE
TERMYSEQUENCE
SOMETHINGADAPTERMYSEQUENCE
```

All the above sequences are trimmed to `MYSEQUENCE` when you use -g ADAPTER. As with 3’ adapters, the resulting read may have a length of zero when the sequence ends with the adapter. For example, the read

***SOMETHINGADAPTER**
will be empty after trimming.

Trimming paired-end reads

Cutadapt supports trimming of paired-end reads, trimming both reads in a pair at the same time.

Assume the input is in `reads.1.fastq` and `reads.2.fastq` and that `ADAPTER_FWD` should be trimmed from the forward reads (first file) and `ADAPTER_REV` from the reverse reads (second file).

The basic command-line is:

```
cutadapt -a ADAPTER_FWD -A ADAPTER_REV -o out.1.fastq -p
out.2.fastq reads.1.fastq reads.2.fastq
```

`-p` is the short form of `--paired-output`. The option `-A` is used here to specify an adapter sequence that cutadapt should remove from the second read in each pair. There are also the options `-G`, `-B`. All of them work just like their lowercase counterparts, except that the adapter is searched for in the second read in each paired-end read. There is also option `-U`, which you can use to remove a fixed number of bases from the second read in a pair.

While it is possible to run cutadapt on the two files separately, processing both files at the same time is highly recommended since the program can check for problems in your input files only when they are processed together.

When you use `-p`/`--paired-output`, cutadapt checks whether the files are properly paired. An error is raised if one of the files contains more reads than the other or if the read names in the two files do not match. Only the part of the read name before the first space is considered. If the read name ends with `/1` or `/2`, then that is also ignored.

To find if the adapters are on the 3' end or 5' end I used the following bash command:

```
zgrep -B 1 -A 2 "AGATCGGAAGAGCACACGTCTGAACTCCAGTCA"
/projects/bgmp/shared/2017_sequencing/demultiplexed/31_4F_fox_S22
_L008_R1_001.fastq.gz | head -n 20
@K00337:83:HJKJNBBXX:8:1101:18456:1226 1:N:0:NATCTTGC+NCAAGATC
GTCGAGAGTTGTAGGCGCGGGACACGGTATTCCAGGCGTCCATGTAGCGGTCCATGCACATGGC*
*AGATCGGAAGAGCACACGTCTGAACTCCAGTCA**CGAT
+
AAFFFJJJFJJJJJJJJJJJJJJJJJJJJJJJFJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJ
JJJJJJJJJJJJJJJJJJJJFJJJJJJJJJJJFF
--
@K00337:83:HJKJNBBXX:8:1101:14935:1244 1:N:0:NATCTTGC+NCAAGATC
CAGGTTCGTCGTCTAGCCAGCCTTCAGGCTTGGTGGCCTCTTCATCTGGG**AGATCGGAAGAGC
ACACGTCTGAACTCCAGTCA**CGATCTTGCATCTCGTAT
+
AAFFFJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJ
JJJJJJJJJJJJFJJJJJJJJJJJJJ-<FJJJJJ
--
@K00337:83:HJKJNBBXX:8:1101:12327:1297 1:N:0:NATCTTGC+NCAAGATC
CTCGGTTATCAACTTCTAGCAGTCGTAGTTCACCAGGTTTTAGGTCGTTTGTTG**AGATCGGAA
GAGCACACGTCTGAACTCCAGTCA**CGATCTTGCATCTC
+
AAFFFJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJFJJJJJJJJJJJJJJJJJJJJJJJJJJJ
JJJJJJJJJJJJJJJJJJJJJJJJJJJJJJAJFJ
--
@K00337:83:HJKJNBBXX:8:1101:14286:1349 1:N:0:NATCTTGC+NCAAGATC
CTCCCGAGTGCTATACATACCGGCATTCATCACCCTGTGCTACCAGCATCTGGTCCCA**AGATC
GGAAGAGCACACGTCTGAACTCCAGTCA**CGATCTTGCA
+
AAFFFJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJFJJAJJJJJFJJJJJFJJJJ
JJJJJJJJJJJJJFFJJJJJJJAFJJJJJJJJFJ-
```

The I used ctrl+F to look up the adapter sequence and see the side where it appears(highlighted with astrics) . It appeared to be on the 3' side in both reads of both files. So

The cutadapt command:

```
cutadapt -a AGATCGGAAGAGCACACGTCTGAACTCCAGTCA -A
AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT -o
./trimmed_reads/trimmed_31_4F_fox_S22_L0
08_R1.fastq.gz -p
./trimmed_reads/trimmed_31_4F_fox_S22_L008_R2.fastq.gz
/projects/bgmp/shared/2017_sequencing/demultiplexed/31_4F_fox_S22
_L008_R1_001.fas
tq.gz
/projects/bgmp/shared/2017_sequencing/demultiplexed/31_4F_fox_S22
_L008_R2_001.fastq.gz
```

```
cutadapt -a AGATCGGAAGAGCACACGTCTGAACTCCAGTCA -A
AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT -o
./cutadapt_adapter_trim/trimmed_10_2G_both_S8_L008_R1_001.fastq.g
z -p
./cutadapt_adapter_trim/trimmed_10_2G_both_S8_L008_R2_001.fastq.g
z /projects/bgm
p/shared/2017_sequencing/demultiplexed/10_2G_both_S8_L008_R1_001.
fastq.gz
/projects/bgmp/shared/2017_sequencing/demultiplexed/10_2G
_both_S8_L008_R2_001.fastq.gz >> cutadapt_report_10-
2G_both_S8_L008.txt
```

Then to confirm that the adapter was actually trimmed from the files, I ran the following command and nothing should appear

```
zgrep -B 1 -A 2 "AGATCGGAAGAGCACACGTCTGAACTCCAGTCA"
trimmed_31_4F_fox_S22_L008_R1.fastq.gz | head
```

*The cutadapt summary report is saved as **cutadapt_report.txt** in **cutadapt_adapter_trim directory**

Trimmomatic

*explanation of parameters:

- LEADING: Cut bases off the start of a read, if below a threshold quality

- TRAILING: Cut bases off the end of a read, if below a threshold quality
- SLIDINGWINDOW: Performs a sliding window trimming approach. It starts scanning at the 5‟ end and clips the read once the average quality within the window falls below a threshold. By considering multiple bases, a single poor quality base will not cause the removal of high quality data later in the read.
- MINLEN: Drop the read if it is below a specified length

Paired End Mode

For paired-end data, two input files, and 4 output files are specified, 2 for the 'paired' output where both reads survived the processing, and 2 for corresponding 'unpaired' output where a read survived, but the partner read did not.

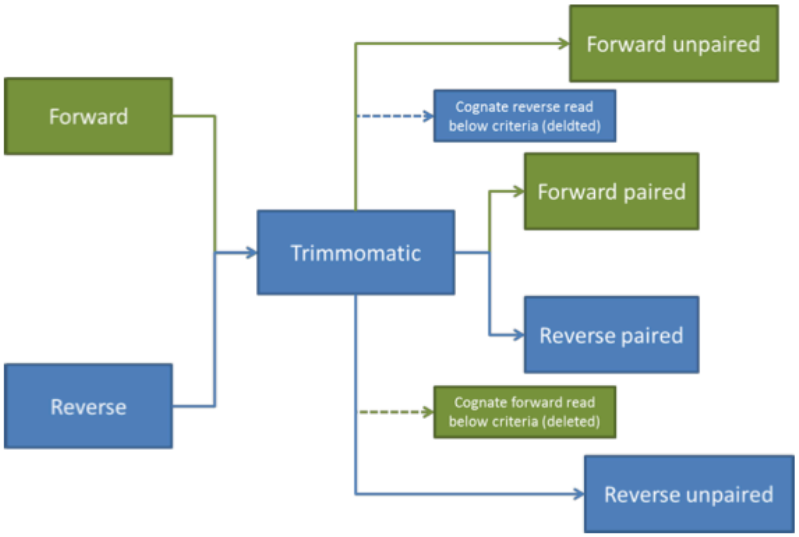

Figure 1: Flow of reads in Trimmomatic Paired End mode

The command was:

```
#!/bin/bash
#SBATCH --account=bgmp
#SBATCH --partition=bgmp
#SBATCH -c 1
#SBATCH --time=0-3

conda activate QAA

/usr/bin/time -v trimmomatic PE -phred33
/projects/bgmp/malm/bioinfo/Bi623/FastQC/cutadapt_adapter_trim/tr
immed_31_4F_fox_S22_L008_R1.fastq.gz \

/projects/bgmp/malm/bioinfo/Bi623/FastQC/cutadapt_adapter_trim/tr
immed_31_4F_fox_S22_L008_R2.fastq.gz \

out_paired_31_4F_fox_S22_L008_R1_001.fastq.gz \

out_unpaired_31_4F_fox_S22_L008_R1_001.fastq.gz \

out_paired_31_4F_fox_S22_L008_R2_001.fastq.gz \

out_unpaired_31_4F_fox_S22_L008_R2_001.fastq.gz \

ILLUMINACLIP:TruSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3
SLIDINGWINDOW:5:15 MINLEN:35



#TruSeq3-PE.fa.... Remove Illumina adapters provided in the
TruSeq3-PE.fa file (provided). Initially

#Trimmomatic will look for seed matches (16 bases) allowing
maximally 2

#mismatches. These seeds will be extended and clipped if in the
case of paired end

#reads a score of 30 is reached (about 50 bases), or in the case
```

```
of single ended reads a

#score of 10, (about 17 bases).
```

```
#!/bin/bash
#SBATCH --account=bgmp
#SBATCH --partition=bgmp
#SBATCH -c 1
#SBATCH --time=0-3

conda activate QAA

/usr/bin/time -v trimmomatic PE -phred33
/projects/bgmp/malm/bioinfo/Bi623/FastQC/cutadapt_adapter_trim/tr
immed_10_2G_both_S8_L008_R1_001.fastq.gz \

/projects/bgmp/malm/bioinfo/Bi623/FastQC/cutadapt_adapter_trim/tr
immed_10_2G_both_S8_L008_R2_001.fastq.gz \

out_paired_10_2G_both_S8_L008_R1_001.fastq.gz \

out_unpaired_10_2G_both_S8_L008_R1_001.fastq.gz \

out_paired_10_2G_both_S8_L008_R2_001.fastq.gz \

out_unpaired_10_2G_both_S8_L008_R2_001.fastq.gz \
ILLUMINACLIP:TruSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3
SLIDINGWINDOW:5:15 MINLEN:35

#TruSeq3-PE.fa.... Remove Illumina adapters provided in the
TruSeq3-PE.fa file (provided). Initially

#Trimmomatic will look for seed matches (16 bases) allowing
maximally 2

#mismatches. These seeds will be extended and clipped if in the
case of paired end

#reads a score of 30 is reached (about 50 bases), or in the case
of single ended reads a

#score of 10, (about 17 bases).
```

***slurm-15937302.out**

```
TrimmomaticPE: Started with arguments:
 -phred33
/projects/bgmp/malm/bioinfo/Bi623/FastQC/cutadapt_adapter_trim/tr
immed_31_4F_fox_S22_L008_R1.fastq.gz
/projects/bgmp/malm/bioinfo/Bi623/FastQC/cutadapt_adapter_trim/tr
immed_31_4F_fox_S22_L008_R2.fastq.gz
out_paired_31_4F_fox_S22_L008_R1_001.fastq.gz
out_unpaired_31_4F_fox_S22_L008_R1_001.fastq.gz
out_paired_31_4F_fox_S22_L008_R2_001.fastq.gz
out_unpaired_31_4F_fox_S22_L008_R2_001.fastq.gz
ILLUMINACLIP:TruSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3
SLIDINGWINDOW:5:15 MINLEN:35
java.io.FileNotFoundException:
/gpfs/projects/bgmp/malm/bioinfo/Bi623/FastQC/trimmomatic/TruSeq3
-PE.fa (No such file or directory)
        at java.base/java.io.FileInputStream.open0(Native Method)
        at
java.base/java.io.FileInputStream.open(FileInputStream.java:213)
        at java.base/java.io.FileInputStream.<init>
(FileInputStream.java:152)
        at
org.usadellab.trimmomatic.fasta.FastaParser.parse(FastaParser.jav
a:54)
        at
org.usadellab.trimmomatic.trim.IlluminaClippingTrimmer.loadSequen
ces(IlluminaClippingTrimmer.java:110)
        at
org.usadellab.trimmomatic.trim.IlluminaClippingTrimmer.makeIllumi
naClippingTrimmer(IlluminaClippingTrimmer.java:71)
        at
org.usadellab.trimmomatic.trim.TrimmerFactory.makeTrimmer(Trimmer
Factory.java:32)
        at
org.usadellab.trimmomatic.Trimmomatic.createTrimmers(Trimmomatic.
java:59)
        at
org.usadellab.trimmomatic.TrimmomaticPE.run(TrimmomaticPE.java:55
2)
```

```
        at
org.usadellab.trimmomatic.Trimmomatic.main(Trimmomatic.java:80)
Input Read Pairs: 3788343 Both Surviving: 3597908 (94.97%)
Forward Only Surviving: 151039 (3.99%) Reverse Only Surviving:
2965 (0.08%) Dropped: 36431 (0.96%)
TrimmomaticPE: Completed successfully
        Command being timed: "trimmomatic PE -phred33
/projects/bgmp/malm/bioinfo/Bi623/FastQC/cutadapt_adapter_trim/tr
immed_31_4F_fox_S22_L008_R1.fastq.gz
/projects/bgmp/malm/bioinfo/Bi623/FastQC/cutadapt_adapter_trim/tr
immed_31_4F_fox_S22_L008_R2.fastq.gz
out_paired_31_4F_fox_S22_L008_R1_001.fastq.gz
out_unpaired_31_4F_fox_S22_L008_R1_001.fastq.gz
out_paired_31_4F_fox_S22_L008_R2_001.fastq.gz
out_unpaired_31_4F_fox_S22_L008_R2_001.fastq.gz
ILLUMINACLIP:TruSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3
SLIDINGWINDOW:5:15 MINLEN:35"
        User time (seconds): 213.67
        System time (seconds): 3.46
        Percent of CPU this job got: 98%
        Elapsed (wall clock) time (h:mm:ss or m:ss): 3:41.27
        Average shared text size (kbytes): 0
        Average unshared data size (kbytes): 0
        Average stack size (kbytes): 0
        Average total size (kbytes): 0
        Maximum resident set size (kbytes): 192288
        Average resident set size (kbytes): 0
        Major (requiring I/O) page faults: 0
        Minor (reclaiming a frame) page faults: 33952
        Voluntary context switches: 6688
        Involuntary context switches: 6044
        Swaps: 0
        File system inputs: 0
        File system outputs: 1440
        Socket messages sent: 0
        Socket messages received: 0
        Signals delivered: 0
        Page size (bytes): 4096
```

```
        Exit status: 0
```

*My comments:
#94.97% of both the forward and reverse reads passed the quality and length filters. These pairs are retained for further analysis.
#3.99% of only the forward read (R1) passed the quality and length filters, while the reverse read (R2) was discarded.
#0.08% of only the reverse read (R2) passed the filters, while the forward read (R1) was discarded
#0.96% of the read pairs, both the forward and reverse reads were discarded because they didn't pass the quality and length thresholds

Generated plots showing read length difference between trimmed R1 and R2 reads for each sample

- I ran FASTQC again on the trimmed files inside the Trimmomatic directory using the same bash script
- To get only the images instead of the html files, I unzipped the second file (besides the html) generated by fastqc run, and inside that file, there is an Images directory, containing all the images. Then i scp those to my local directory (QAA) to be added to the R markdown.

Part 3: Alignment and strand-specificity

Installed star, numpy, matplotlib, and htseq

In my QAA conda environment:

```
conda install <software>
```

- Ran star (db and alignment)
- Counted mapped and unmapped reads using script from PS8 (Just added argsparse):
  File 1 output

```
The number of mapped reads is **6969863**
The number of unmapped reads is **225953**

        Command being timed: "python ./sam_parsing.py -f
alignment_output_31_4F_fox_S22_L008Aligned.out.sam"
        User time (seconds): 6.57
        System time (seconds): 1.02
        Percent of CPU this job got: 89%
        Elapsed (wall clock) time (h:mm:ss or m:ss): 0:08.47
        Average shared text size (kbytes): 0
        Average unshared data size (kbytes): 0
        Average stack size (kbytes): 0
        Average total size (kbytes): 0
        Maximum resident set size (kbytes): 10824
        Average resident set size (kbytes): 0
        Major (requiring I/O) page faults: 0
        Minor (reclaiming a frame) page faults: 2535
        Voluntary context switches: 270
        Involuntary context switches: 20
        Swaps: 0
        File system inputs: 0
        File system outputs: 0
        Socket messages sent: 0
        Socket messages received: 0
        Signals delivered: 0
        Page size (bytes): 4096
        Exit status: 0
```

File 2output

```
The number of mapped reads is **152719439**
The number of unmapped reads is **2322367**

        Command being timed: "python ./sam_parsing.py -f
alignment_output_10_2G_both_S8_L008Aligned.out.sam"
        User time (seconds): 144.74
        System time (seconds): 13.13
        Percent of CPU this job got: 98%
        Elapsed (wall clock) time (h:mm:ss or m:ss): 2:40.08
        Average shared text size (kbytes): 0
        Average unshared data size (kbytes): 0
        Average stack size (kbytes): 0
        Average total size (kbytes): 0
        Maximum resident set size (kbytes): 14356
        Average resident set size (kbytes): 0
        Major (requiring I/O) page faults: 0
        Minor (reclaiming a frame) page faults: 2546
        Voluntary context switches: 1234
        Involuntary context switches: 391
        Swaps: 0
        File system inputs: 0
        File system outputs: 0
        Socket messages sent: 0
        Socket messages received: 0
        Signals delivered: 0
        Page size (bytes): 4096
        Exit status: 0
        ```
```

Now working on **htseq-count** to get the counts that mapped to a feature. Ran it twice using two different flags to determine strand specificity

I wrote the following bash script (just changed file names for sample 2):
```

```
#!/bin/bash
#SBATCH --account=bgmp
#SBATCH --partition=bgmp
#SBATCH -c 1
#SBATCH --time=0-3

conda activate QAA


/usr/bin/time -v htseq-count --stranded=yes
alignment_output_31_4F_fox_S22_L008Aligned.out.sam
../Mus_musculus.GRCm39.112.gtf >
htseq_count_stranded_131_4F_fox_S22_L008.out

/usr/bin/time -v htseq-count --stranded=reverse
alignment_output_31_4F_fox_S22_L008Aligned.out.sam
../Mus_musculus.GRCm39.112.gtf >
htseq_count_reverse_31_4F_fox_S22_L008.out
```

*Stranded=yes means that the RNA seq library is strand specific, but we got less counts with the flag **--stranded=yes** compared to the flag **--stranded=reverse** which means that the library is NOT strand specific.

To get the total number of genes and the number mapped genes from htseq-count output I used the bash command from ICA4

*Total counts :

```
cat htseq_count_reverse_31_4F_fox_S22_L008.out | cut -f 2 | awk
'{sum+=$1} END {print sum}'

# For FOX sample:
3597908

# For 10-2g sample
77520903
```

*Counts in stranded=True vs Stranded=reverse

```
# For Fox Sample

grep -v "^__" htseq_count_stranded_131_4F_fox_S22_L008.out| cut -
f 2 | awk '{sum+=$1} END {print sum}'

> 180913 (5.02%)

grep -v "^__" htseq_count_reverse_31_4F_fox_S22_L008.out | cut -f
2 | awk '{sum+=$1} END {print sum}'

> 2957009 (82.19%)

# For 10_2g sample
> 2957148 (3.81%)
> 67359730 (86.90%)
```