In-class exercise: Working with Illumina sequencing reads

Log onto a computing node before beginning this exercise.

- Perform the operations below first using the week4 sequence files.
- Then repeat the operations using the Day data files in the shared folder.

You've downloaded and uncompressed a set of sequencing files into their own directory. The original file ended in tar.gz. Now the various files that resulted from uncompressing end in:
.fastq.gz

Write a shell script that:

- unzips each file but leaves the original copy intact

```
for filename in *.tiny.fastq.gz
do

  base=$(basename $filename .tiny.fastq.gz)
  echo ${base}

  gunzip -c ${base}.tiny.fastq.gz > ${base}.tiny.fastq

done
```

Now the directory has two sets of files, each with a different extension. Make a new directory and move all the unzipped files to it.

```
mkdir unzip_files
mv *.fastq unzip_files/
```

But wouldn't it have been more efficient to include this step in the original shell script? Let's try that. First, remove the directory that contains the unzipped files.

```
rm -r unzip_files/
```

Then go back to the shell script and amend it to include commands to:

- make a new directory
- place the unzipped files there

```
for filename in *.tiny.fastq.gz
do

  base=$(basename $filename .tiny.fastq.gz)
  echo ${base}

  gunzip -c ${base}.tiny.fastq.gz > ${base}.tiny.fastq

done
```

```
# check if the directory exists, if not, create it
if [ ! -d "unzip_files" ]; then
  mkdir -p "unzip_files"
fi

# move files into the new directory
echo "moving files into unzip_files..."
mv /courses/BIOL3411.202430/students/bolduc.je/assigments/Unit2/2C_deliv *tiny.fastq
"unzip_files"/

echo "Files moved successfully into unzip_files."
```

Use the `grep` command to determine the number of reads in one unzipped file

```
grep -c '^@' SRR6805880.tiny.fastq
```

then in all the unzipped files, printing to the screen

```
grep -c '^@' *.fastq
```

then in all the unzipped files, printing to a new file.

```
nano num_reads.txt
```

Use head to examine the start of each read and see if there is an obvious adapter sequence to trim. Every read starts with the sequence “TGCAG”

Now you want to generate quality control reports for each of the files, using fastqc/0.11.9. This package requires that you first load the module OpenJDK/19.0.1, and the command operates on zipped files (ending in fastq.gz).
First test out the operation on only one file, using the command line.

Now try performing this task on all the files, in two different ways-- using a shell script and using a bash script.

Shell Script:

```
for filename in *.tiny.fastq.gz
do

  base=$(basename $filename .tiny.fastq.gz)
  echo ${base}

  fastqc ${base}.tiny.fastq.gz > ${base}.tiny_fastq.html

done
```

Bash Script:

```
#!/bin/bash
#SBATCH --partition=short
#SBATCH --job-name=QualReport
#SBATCH --time=24:00:00
#SBATCH --nodes=1
#SBATCH --cpus-per-task=2
#SBATCH --mem=256G
#SBATCH --output=%j.output
#SBATCH --error=%j.error

cd /courses/BIOL3411.202430/students/bolduc.je/assignments/Unit2/2C_Deliv
fastqc *.tiny.fastq.gz
```

After generating the new files, make a separate directory and put the .html files into it (you can try adding this step directly to your script from above, if you like).

Open one of the .html files (you'll need to do this through the OOD, using a browser to view it).

Watch this video on interpreting the report.