

Nombre y Apellido: Juan Carlos García Estupiñán

Actividad 2.- Manipulación y formateo de archivos: Formato FASTQ y FASTA

Objetivo

El propósito de esta actividad es desarrollar un flujo de trabajo completo (denominado en inglés *pipeline*) bioinformático que nos permita procesar una serie de datos biológicos. Con esta actividad, se pretende que el estudiante adquiera destrezas para interactuar con el sistema operativo a través de la línea de comandos y que sea capaz de desarrollar Shell scripts propios para resolver diferentes retos bioinformáticos focalizados en dos tipos de formato de texto, el formato FASTQ y en formato FASTA.

Parte I: Creación del directorio de trabajo.

Para inicializar este ejercicio, deberán crear y organizar un nuevo directorio de trabajo que contenga los puntos más importantes que vamos a estudiar. La idea de esta organización es que alguien que no esté familiarizado con su proyecto debería poder mirar los archivos almacenados en el ordenador y comprender en detalle lo que hicimos y por qué lo hicimos. Este «alguien» podría ser cualquiera de una variedad de personas: alguien que leyó el artículo que hemos publicado y quiere intentar reproducir nuestro trabajo, un colaborador que quiere comprender los detalles de los experimentos realizados, un futuro estudiante que trabaja con ustedes en el laboratorio y deseamos ampliar el trabajo hecho hasta el momento, un asesor de investigación que puede estar interesado en comprender nuestro trabajo o que puede estar evaluando nuestras habilidades de investigación. Sin embargo, lo más común es que ese "alguien" seamos nosotros mismos.

Usando comandos de Linux a través de un terminal, deberá crear la siguiente estructura de directorios para este proyecto que identificará de la siguiente manera:

User_project_year_publication donde:

- User corresponde a su apellido.
- project corresponde al nombre de su proyecto hipotético en el cual se encuentra trabajando (por ejemplo: humanTonsils)
- year es el año de la publicación o investigación que está realizando.
- Publication corresponde a una revista o conferencia en donde quiere publicar sus resultados.

Este directorio, en mi caso, ***Soler_humanTonsils_2022_Nature*** será creado dentro del directorio ***Documents*** que a su vez contendrá 5 directorios (***analysis***, ***code***, ***data***, ***tools*** y ***submission***) y dos archivos (***README*** y ***LICENSE***). El directorio ***data*** tiene a su vez los directorios ***raw*** y ***processed***. El directorio ***submission*** contendrá a su vez dos directorios (***version1*** y ***version2***).

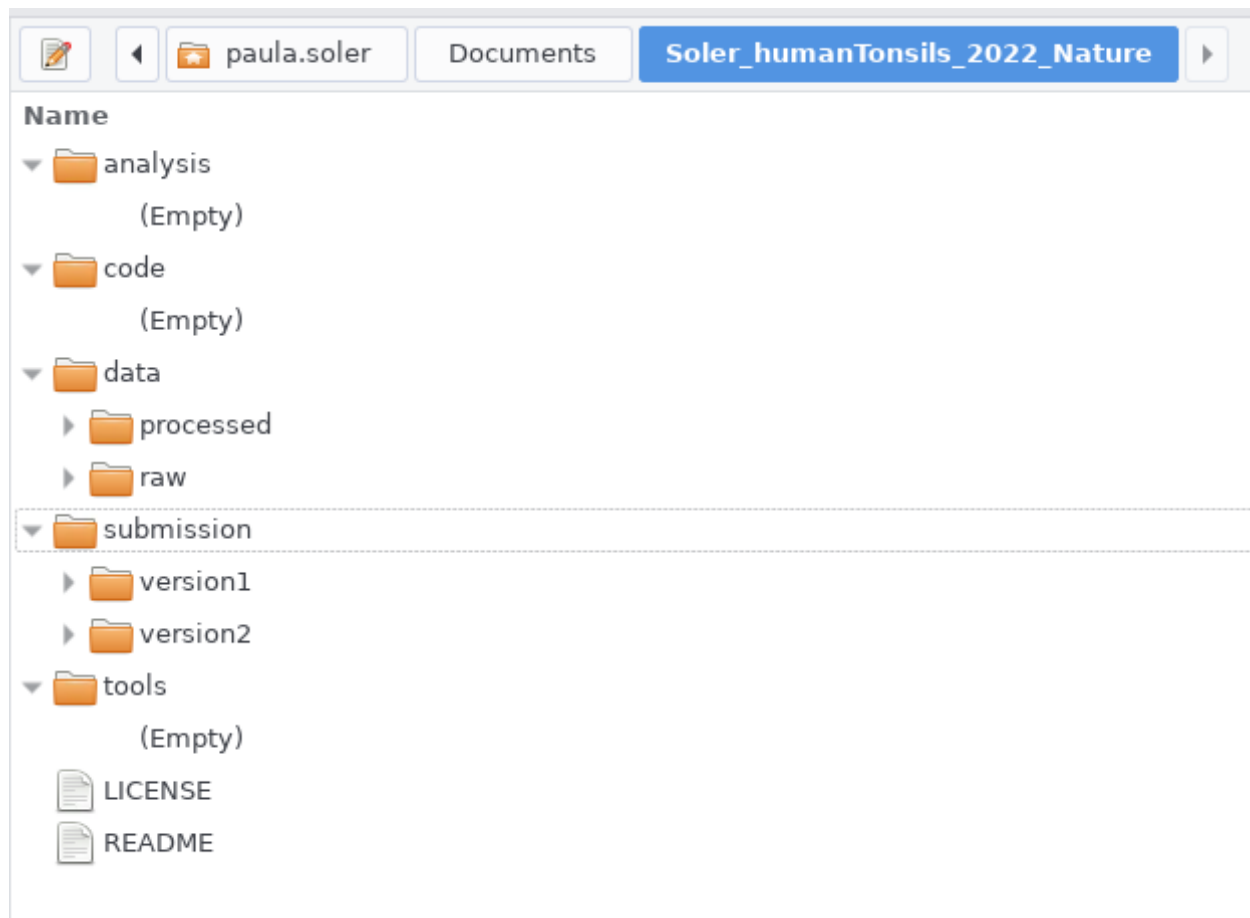


Figura 1. Estructura de directorio vista desde la interfaz gráfica

Después de crear la estructura de directorios antes indicada, incluya una captura de pantalla usando el comando **tree** desde el terminal. También puede incluir una captura de pantalla desde la interfaz gráfica. (0,25 pts)

```
File Edit View Search Terminal Help
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo Garcia_SnakeViral_2022_Nature]$ tree
.
├── analysis
├── code
├── data
│   ├── processed
│   └── raw
├── LICENSE
├── README.md
├── submission
│   ├── version1
│   └── version2
└── tools

9 directories, 2 files
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo Garcia_SnakeViral_2022_Nature]$
```

Parte II: Obtención de datos

Ahora, va a obtener el conjunto de datos con los cuales va a trabajar. En este caso, se trabajará con un conjunto de datos generados mediante la conocida técnica de secuenciación “shotgun” del hígado de una *Boa constrictor* a la que se le diagnosticó la enfermedad del cuerpo de inclusión o IBD como consecuencia de una infección viral.

Para obtener el conjunto de datos, abra un determinado navegador y acceda al enlace que se le determina a continuación: <https://www.ncbi.nlm.nih.gov/pubmed/25993603>. Desplácese hacia abajo y busque la sección “*Related Information*” (esquina inferior derecha) y haga clic en el enlace “SRA” (Short Read Archive). Esto muestra los conjuntos de datos SRA asociados a esta publicación.

Responde a la siguiente pregunta relacionado con el SRA. ¿Qué es y para qué se utiliza? (0,25 pts)

- El SRA se trata de los archivos principales del NIH de datos de secuenciación de alto rendimiento, formando parte de la Colaboración internacional de bases de datos de secuencias de nucleótidos (INSDC) que incluye el Archivo de lectura de secuencias (SRA) del NCBI, el instituto Europeo de Bioinformática (EBI) y la Base de datos de ADN de Japón. Los datos de Sequence Read Archive (SRA, por lo visto antes se les llamaba **short read archives**), disponibles a través de múltiples proveedores en la nube y servidores NCBI, son el repositorio público más grande disponible de datos de secuenciación de alto rendimiento. El archivo acepta datos de todas las ramas de la vida. SRA almacena datos de secuenciación sin procesar e información de alineación para mejorar la reproducibilidad y facilitar nuevos descubrimientos a través del análisis de estos datos (1).

Se le abrirá una ventana nueva donde podrá observar múltiples resultados asociados a todas las muestras procesada. De todas ellas, busque y acceda a “**shotgun sequencing of tissue from snake_6_viral**”. Observe en la parte inferior de la página el número de *run* (SRR #) para este conjunto de datos (**SRR1984406**). **Añada una captura de pantalla que verifique que ha encontrado este conjunto de datos (0,25 pts)**



Figura 2: Conjunto de datos de snake_6_viral.

Con este número, ya tiene la posibilidad de descargar los datos de archivo de SRA. Para realizarlo, deberá emplear una toolkit específica para descargar, forma automática, los datos asociados a este experimento en su entorno virtual. Para ello deberán:

1. Instalar SRA toolkit

A continuación, se le expone los pasos claves para la instalación de esta herramienta. Todos ellos han sido extraídos de: <https://github.com/ncbi/sra-tools/wiki/02.-Installing-SRA-Toolkit>

- Descargue la herramienta **SRA toolkit** en el directorio creado anteriormente llamado **tools** (que utilizará para guardar las herramientas empleadas) utilizando el siguiente comando:

```
wget --output-document sratoolkit.tar.gz https://ftp-trace.ncbi.nlm.nih.gov/sra/sdk/current/sratoolkit.current-centos_linux64.tar.gz
```

- Extraiga el contenido del archivo comprimido con el siguiente comando:

```
tar -vxzf sratoolkit.tar.gz
```

- Una vez descomprimido, puede eliminar el archivo descargado. Dentro del directorio generado llamado **sratoolkit.3.0.0-centos_linux64**, podrá encontrar otro directorio llamada **bin** y dentro de éste podrá observar una gran cantidad archivos ejecutables.

Responde a la siguiente pregunta: ¿Cuántos archivos ejecutables puede diferenciar dentro del directorio bin? Recuerde añadir una captura de pantalla con los comandos empleados para contar el número de archivos ejecutables y repórtelo por pantalla (0,25 pts)

```
File Edit View Search Terminal Help
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo bin]$ find . -type f -perm -u+rx | wc -l
39
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo bin]$
```

- Dentro del directorio **bin**, localice el ejecutable **fastq-dump** que será aquel que utilizará para descargar de forma automática los archivos asociados al SRA. Para ejecutarlo de forma sencilla, sin necesidad de que cada vez que lo quiera ejecutar tenga que determinar toda la ruta completa, puede añadir la ruta completa del mismo a la variable de entorno PATH de la siguiente manera:

```
export PATH=$PATH:$PWD/sratoolkit.3.0.0-centos_linux64/bin
```

- Verifique que **fastq-dump** puede ser encontrado por el Shell con el siguiente comando y adjunte una captura de pantalla del resultado del mismo (0.5 pts).

```
which fastq-dump
```

```
File Edit View Search Terminal Help
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo tools]$ which fastq-dump
~/Desktop/Garcia_SnakeViral_2022_Nature/tools/sratoolkit.3.0.0-centos_linux64/bin/fastq-dump
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo tools]$
```

- Finalmente, escriba el comando **fastq-dump** en la terminal. Cuando esto lo realice por primera vez, le aparecerá un texto que le indicará que la instalación no está configurada y para ello deberá correr el siguiente comando:

```
vdb-config --interactive
```

- Verá una pantalla azul con diferentes opciones, donde deberá habilitar la opción **Enable Remote Access** (quizá por defecto la vea marcada) y darle a la tecla X para salir de este menú interactivo.

2. Empleo de SRA toolkit para descargar las lecturas

Finalmente, descargue el conjunto de datos utilizando el comando “fastq-dump” seguido del número SRR # del conjunto de datos seleccionados. Añada la opción --split-files para crear 2 archivos sincronizados por cada una de las lecturas emparejadas (paired-end reads). Aquí se muestra el comando completo a ejecutar:

```
fastq-dump SRR1984406 --split-files
```

Confirme que descargó los archivos correctamente añadiendo una captura de pantalla del comando **ls -l** en el directorio **data/raw** (donde deberá localizar estos dos archivos) (0.25 pts).

```

File Edit View Search Terminal Help
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo tools]$ which fastq-dump
~/Desktop/Garcia_SnakeViral_2022_Nature/tools/sratoolkit.3.0.0-centos_linux64/bin/fastq-dump
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo tools]$ fastq-dump SRR1984406 --split-files
Rejected 1263 READS because READLEN < 1
Read 8991 spots for SRR1984406
Written 8991 spots for SRR1984406
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo tools]$ ls
sratoolkit.3.0.0-centos_linux64 sratoolkit.tar.gz SRR1984406_1.fastq SRR1984406_2.fastq
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo tools]$ mv SRR* ../data/raw/
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo tools]$ █

```

- En su momento me faltó hacer el ***ls -l***, lo muestro por separado:

```

File Edit View Search Terminal Help
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo raw]$ ls -l
total 5360
-rw-r--r-- 1 UNIVERSIDADVIU\juan.carlos UNIVERSIDADVIU\domain users 2707290 Nov 15 15:30 SRR1984406_1.fastq
-rw-r--r-- 1 UNIVERSIDADVIU\juan.carlos UNIVERSIDADVIU\domain users 2779914 Nov 15 15:30 SRR1984406_2.fastq
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo raw]$

```

Seguidamente, responda a cada una de las preguntas que se le indican, adicionando siempre los comandos empleados en cada caso y una captura de pantalla de ellos junto a su resultado.

- **¿Qué tipo de formato tienen los archivos descargados y para qué se utiliza dicho formato? (0.25 pts).**
 - EL formato FASTQ se trata de una secuenciación basada en texto que almacena archivos de datos con los contenidos de datos crudos de secuencia y el puntaje de calidad (2).
 - A su vez se encuentra formado por los siguientes componentes (3):
 - Un identificador de secuencia.
 - La secuencia (los nucleótidos ACTG y N en caso de valores no identificados).
 - Un separador, al cual simplemente se le añade el signo “+”.
 - El puntaje de calidad de la base. Estos están codificados por Phred +33, usando caracteres de ASCII que representan los valores numéricos de los puntajes de calidad.
- **¿Cuántas líneas en total tienen cada uno de los archivos? (0,5 pts)**
 - **Script: 1nrow.sh.**

```

File Edit View Search Terminal Help
1  #!/bin/bash
    1
    2  ## En mi caso los datos son las réplicas de SRR1984406
    3  ## en formato fastq
    4
    5  ## EJECUCIÓN ##
    6  echo ">>> Vemos el número de líneas de cada archivo"
    7  wc -l $1 $2 | head -2
    8
~

```

- **Stdout: 1nrow.sh.** (además creo las variables fuera del script srr1 y srr2 para que se vean mejor las imágenes y no sea tan larga).

```

File Edit View Search Terminal Help
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$ srr1=./data/raw/SRR1984406_1.fastq
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$ srr2=./data/raw/SRR1984406_2.fastq
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$ ./1nrow.sh $srr{1..2}
>>> Vemos el número de líneas de cada archivo
32984 ./data/raw/SRR1984406_1.fastq
33892 ./data/raw/SRR1984406_2.fastq
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$

```

- ¿Cuántas secuencias incluye el archivo **SRR1984406_1.fastq**? (0,5 pts)

- **Script: 2nsecuencias.sh**

```

File Edit View Search Terminal Help
1  #!/bin/bash
    1
    2  ## En mi caso $1 es la primera réplica de SRR1984406
    3
    4  ## Como en un archivo .fastq una secuencia se toma cada cuatro
    5  ## líneas, habrá que dividir el número total de filas entre 4
    6  echo ">>> Número de secuencias del formato .fastq"
    7  num_linea=$(echo $(awk {'print NR'} $1 | tail -1))
    8  echo $((num_linea/4))
    9
~

```

- **Stdout: 2nsecuencias.sh**

```

File Edit View Search Terminal Help
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$ ./2nsecuencias.sh $srr1
>>> Número de secuencias del formato .fastq
8246
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$

```

- ¿Cuántas veces aparece la secuencia ATGATG en cada uno de los archivos descargados? (0,5 pts)

- Script: 3seq_identificar.sh.

```
File Edit View Search Terminal Help
1 #!/bin/bash
2 ## Usaré ambas réplicas de SRR1984406
3
4 ## EJECUCIÓN ##
5 ## Usamos grep -o, por que da una lista de las repeticiones, y
6 ## entiendo que si da la casualidad de que repite más de una ve
7 ## en una línea también lo reporta
8 echo ">>> Vemos las repeticiones de ATGATG en $1"
9 grep -o "ATGATG" $1 | wc -l
10 echo ">>> Vemos las repeticiones de ATGATG en $2"
11 grep -o "ATGATG" $2 | wc -l
12 ## También se podría usar:
13 # grep -o "ATGATG" $1 | uniq -c
14 # grep -o "ATGATG" $2 | uniq -c
15
```

- Stdout: 3seq_identificar.sh.

```
File Edit View Search Terminal Help
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$ ./3seq_indentificar.sh $srr{1..2}
>>> Vemos las repeticiones de ATGATG en ../data/raw/SRR1984406_1.fastq
835
>>> Vemos las repeticiones de ATGATG en ../data/raw/SRR1984406_2.fastq
842
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$
```

A continuación, va a convertir el archivo **SRR1984406_1.fastq** que se presenta en formato fastq, en otro archivo con un formato un tanto distinto, el formato FASTA.

¿Qué diferencias hay entre un archivo en formato FASTQ y un archivo en formato FASTA? (0,5 pts)

- El formato FASTA estaría únicamente formado por el encabezado con la identificación de la secuencia y la secuencia en sí. No se le añade la tercera y cuarta línea del formato fastq (es decir, el separador con el signo "+" y los puntajes de calidad). Además, el encabezado en vez de empezar con un "@" empezaría con un ">".

Para ello, deberá seleccionar la primera y la segunda línea de cada una de las 4 líneas que conforman cada una de las lecturas o *reads* del archivo **SRR1984406_1.fastq** y así confeccionar un archivo final llamado **all_sequences.fasta** que contenga el *header* o cabecera y la secuencia asociada a cada lectura. Recuerda guardar este archivo en el directorio **data/processed**.

Visualice las 5 primeras líneas del archivo creado en el directorio determinado. Incluya una captura de pantalla con el código empleado y el resultado solicitado (0,5 pts)

- Script: `fasta_format.sh`.

```
File Edit View Search Terminal Help
1  #!/bin/bash
2  ## Usaré la primera réplica de SRR1984406
3
4  ## VARIABLES ##
5  ## Yo abajo ../data/processed/all_sequences.fasta
6  read -p "Almacenar todas las secuencias fasta: " fasta
7
8  ## EJECUCIÓN ##
9  ## Convertimos el formato fastq a fasta, para ello usamos sed. La
10 ## idea primero es sustituir las "@" cada cuatro líneas por ">"
11 ## y luego seleccionar únicamente la primera y la segunda (encabezado
12 ## y secuencia), así cada cuatro líneas a su vez.
13 sed -n '1~4s/^@/>/p;2~4p' $1 > $fasta
14 head -5 $fasta
15
```

- Stdout: `fasta_format.sh`.

```
File Edit View Search Terminal Help

[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$ ./4fasta_format.sh $srr1
Almacenar todas las secuencias fasta: ../data/processed/all_sequences.fasta
>SRR1984406.1 1 length=135
GACGACTGCCATCTGAACGTGTGGAATCAACGGAGCCACATCTGACTTCCAGTATCCATCCGAAGTTCTCCATTCAATAG
TGAGGAATCTGACGACTGCCATCTGAACGTGTGGAATCAACGGAGCCACATCTGA
>SRR1984406.2 2 length=134
TTTGGAATTTCTGTATCCATCCGAAGTTCTCCATTCAATAGTGAGGAATCTGACGACTGCCATCTGAACGTGTGGAAT
CAACGGAGCCACATCTGACTAGTGCCAGCATGAGCGACTCCACGCCATTGGG
>SRR1984406.3 3 length=134
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$
```

Una vez creado, seleccione aleatoriamente 5 lecturas (con su cabecera y su secuencia asociada) del archivo `all_sequences.fasta` y seguidamente guárdelas en 5 archivos de texto distintos denominados `secuencia1.fasta`, `secuencia2.fasta` y así sucesivamente. Cada uno de ellos contendrá lo siguiente:

```
>SRR1984406.1 1 length=135
```

```
GACGACTGCCATCTGAACGTGTGGAATCAACGGAGCCACATCTGACTTCCAGTATCCATCCGAAGTTCTCCATTCA
ATAGTGAGGAATCTGACGACTGCCATCTGAACGTGTGGAATCAACGGAGCCACATCTG
```

Muestre los archivos creados adjuntado una captura de pantalla (0,5 pts)

- Script: `5randomseq.sh`.

```

File Edit View Search Terminal Help
1  #!/bin/bash
2  ## VARIABLES ##
3  fasta=$1 # all_sequences.fasta en los datos procesados
4  header_seq_array=($2 $3 $4 $5 $6) # las 5 secuencias a crear
5  ## Lo siguiente una variable para un encabezado aleatorio:
6  read -p "Almacenar el encabezado aleatorio: " random_header
7
8  ## EJECUCIÓN ##
9  ## Hacemos el siguiente bucle for para crear cada uno de los archivos.
10 ## En sí, la primera parte selecciona un header aleatorio, y la segunda parte,
11 ## apartir de ese valor, selecciona también la siguiente línea (su secuencia)
12 ## y omite las demás, así hasta las 5 seq.
13
14 for HEADER_SEQ in ${header_seq_array[*]}
15 do
16     grep '^>' $fasta | shuf -n 1 > $random_header
17     grep -Fwf $random_header -A 1 $fasta | grep -v '^--$' > $HEADER_SEQ
18 done
19 rm $random_header

```

- Stdout: 5randomseq.sh.

```

File Edit View Search Terminal Help
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$ all_fasta=./data/processed/all_sequences.fasta
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$ ./5randomseq.sh $all_fasta ../data/processed/secuencia{1..5}.fasta
Almacenar el encabezado aleatorio: ../data/processed/random_header.txt
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$ ls -l ../data/processed/
total 1344
-rw-r--r-- 1 UNIVERSIDADVIU\juan.carlos UNIVERSIDADVIU\domain users 1353645 Nov 18 11:33 all_sequences.fasta
-rw-r--r-- 1 UNIVERSIDADVIU\juan.carlos UNIVERSIDADVIU\domain users 167 Nov 18 11:41 secuencia1.fasta
-rw-r--r-- 1 UNIVERSIDADVIU\juan.carlos UNIVERSIDADVIU\domain users 168 Nov 18 11:41 secuencia2.fasta
-rw-r--r-- 1 UNIVERSIDADVIU\juan.carlos UNIVERSIDADVIU\domain users 168 Nov 18 11:41 secuencia3.fasta
-rw-r--r-- 1 UNIVERSIDADVIU\juan.carlos UNIVERSIDADVIU\domain users 165 Nov 18 11:41 secuencia4.fasta
-rw-r--r-- 1 UNIVERSIDADVIU\juan.carlos UNIVERSIDADVIU\domain users 168 Nov 18 11:41 secuencia5.fasta
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$

```

Con uno de estos archivos creados (puede seleccionar el que prefiera de los 5), deberá realizar un script en BASH (esté deberá estar almacenado en el directorio *code*) que sea capaz de leer dicho archivo, específicamente su secuencia, y deberá reportar por la salida estándar si se trata de una secuencia nucleotídica o aminoacídica.

Pegue el script generado y adicione una captura de pantalla de su ejecución por terminal (2 pts)

- Script: 6seq1.sh.

```

File Edit View Search Terminal Help
1  #!/bin/bash
2  ## VARIABLES ##
3  ## Elegir una de las cinco secuencias
4  read -p "Escoja alguna de sus secuencias: " secuencia_elegida
5  ## Archivo que contiene una lista con la información de la secuencia
6  read -p "Acrhivo que contiene el tipo de secuencia: " tipo_seq
7
8  secuencia=$(grep -v '>' $secuencia_elegida)
9  ## Hacemos el segiente bucle con el condicional dentro. Crea una lista
10 ## de las bases nitrogenadas según una condición
11 for (( i=0; i<${#secuencia}; i++ ))
12 do
13     ## La condición dice que si están una letra u otra, que coincida
14     ## con un nucleótido, dirá posible nucleótido (N por pérdidas)
15     if [[ ${secuencia:$i:1} == A || ${secuencia:$i:1} == G ||
16         ${secuencia:$i:1} == T || ${secuencia:$i:1} == C ||
17         ${secuencia:$i:1} == N ]]
18     then
19         echo "Posible nucleótido"
20     else
21         echo "Letra de aminoácido"
22     fi
23 done > $tipo_seq
24 ## Corroboramos con el siguiente condicional, si hay una línea
25 ## Cabría esperar que se tratan de nucleótidos, si hay 2, habría
26 ## otras letras además de estas que serían aminoácidos.
27 corroboracion=$(uniq $tipo_seq | wc -l)
28 if [ $corroboracion -eq 1 ]
29 then
30     echo ">>> Se trata de una secuencia de nucleótidos"
31 elif [ $corroboracion -gt 1 ]
32 then
33     echo ">>> Se trata de una secuencia de aminoácidos"
34 else
35     "ERROR"
36 fi
37 rm $tipo_seq

```

- Stdout: 6seq1.sh.

```

File Edit View Search Terminal Help
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$ ./6seq1.sh
Escoja alguna de sus secuencias: ../data/processed/secuencial.fasta
Acrhivo que contiene el tipo de secuencia: ../data/processed/tipo_seq.txt
>>> Se trata de una secuencia de nucleótidos
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$

```

A continuación, implemente otro script en BASH (esté deberá estar almacenado en el directorio *code*) que lea cada uno de los 5 archivos generados (secuencia1.fasta, secuencia2.fasta, etc.) y transforme su secuencia de ADN a ARN (recuerde que base nitrogenada cambia entre estas dos moléculas). Una vez traducidas, deberá contar la frecuencia de aparición de cada una de las 4 bases nitrogenadas y reportarlas por la salida estándar para cada una de las 5 secuencias analizadas.

Pegue el script generado y adicione una captura de pantalla de su ejecución por terminal (2 pts)

- Script: 7dna_to_rna.sh.


```
File Edit View Search Terminal Help
1  #!/bin/bash
2  1
3  ## VARIABLES ##
4  ## Secuencias a usar: de la 1 a la 5
5  seq1=$1;seq2=$2;seq3=$3;seq4=$4;seq5=$5
6  read -p "Almacenar las secuencias de rna: " secuencia_rna
7  ## EJECUCIÓN ##
8  ## Usamos un cat para mostrar las 5 secuencias (geader y secuencia)
9  ## y acto seguido usaremos sed para sustituir cada 2 líneas, empezando
10 ## por la segunda (las secuencias), las Timinas por Uracilo.
11 cat $seq{1..5} | sed '2~2s/T/U/g' > $secuencia_rna
12
13 ## Ahora lo que vamos a hacer es contar la aparición de cada una
14 ## de las bases nitrogenadas. La idea es primero filtrar el encabezado
15 ## y se printe primero, y luego crear una lista de los nucleótidos con
16 ## cada uno de los recuentos printandose después. El bucle nos permite
17 ## hacer esto línea por línea con nuestros datos de las secuencias de
18 ## rna, habiéndose de alimentar al bucle.
19 while read linea
20 do
21 echo $linea | grep '>'
22 echo $linea | grep -v '>' | grep -o "[ACGU]" | sort | uniq -c
23 done < $secuencia_rna
~
```

- Stdout: 7dna_to_rna.sh.

```
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$ ./7dna_to_rna.sh ../data/processed/secuencia{1..5}.fasta
Almacenar las secuencias de rna: ../data/processed/rna_secuencia.fasta
>SRR1984406.2835 2835 length=134
40 A
24 C
27 G
43 U
>SRR1984406.5955 5955 length=134
36 A
32 C
30 G
36 U
>SRR1984406.1377 1377 length=134
35 A
40 C
21 G
38 U
>SRR1984406.2911 2911 length=135
48 A
19 C
27 G
41 U
>SRR1984406.4139 4139 length=134
22 A
37 C
38 G
37 U
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo code]$ █
```

Seguidamente debe completar el archivo de texto llamado **LICENSE**, donde deberá indicar que tipo de licencia quiere utilizar para limitar o no el uso, la modificación y la distribución de su código BASH desarrollado. Es muy importante comenzar a concienciarse sobre las posibles licencias que podemos otorgarles a nuestros scripts, por ello, busque información en la red sobre los distintos tipos y cuál se adaptaría mejor a sus preferencias. **Adjunte una captura de pantalla con el contenido de este archivo (0,5 pts).**

- Yo en mi cuenta de GitHub uso siempre, o al menos hasta el momento, la licencia **MIT** (*Massachusetts Institute of Technology*).
 - Consiste en una licencia muy permisiva, que permite el uso de mi código, para muchas cosas (reproducir el experimento, sacar rédito económico...), con las limitaciones de garantía y responsabilidad con este y como única condición, el aviso de derechos de autor del código y hasta donde tengo entendido, el código modificado también tiene que presentar la licencia MIT.
 - En GitHub a la hora de crear un repositorio, es una de las opciones de serie de la página.
 - Adjunto la información que GitHub pone por defecto:

 Juankkar/Git_Iris is licensed under the
MIT License

A short and simple permissive license with conditions only requiring preservation of copyright and license notices. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

Permissions	Limitations	Conditions
✓ Commercial use	✗ Liability	① License and copyright notice
✓ Modification	✗ Warranty	
✓ Distribution		
✓ Private use		

Figura 4. Información resumida de la Licencia MIT.

Executable File	21 lines (17 sloc)	1.04 KB
-----------------	--------------------	---------

```

1 MIT License
2
3 Copyright (c) 2022 Juan Carlos
4
5 Permission is hereby granted, free of charge, to any person obtaining a copy
6 of this software and associated documentation files (the "Software"), to deal
7 in the Software without restriction, including without limitation the rights
8 to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
9 copies of the Software, and to permit persons to whom the Software is
10 furnished to do so, subject to the following conditions:
11
12 The above copyright notice and this permission notice shall be included in all
13 copies or substantial portions of the Software.
14
15 THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
16 IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
17 FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
18 AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
19 LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
20 OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
21 SOFTWARE.

```

Figura 4. Contenido de la licencia MIT.

Además, deberá completar el archivo **README**, explicando en él, el contenido de cada uno de los directorios creados en el proyecto, especificando las características más relevantes de cada uno de ellos. En este archivo puede añadir la información más relevante que considere. **Adjunte una captura de pantalla con el contenido de este archivo (0,25 pts).**

En este apartado lo que he hecho es, añadir el título, mis credenciales, así como copiar el objetivo de la actividad y de explicar cada uno de los directorios, en formato Markdown.

Como no puedo añadir una captura de pantalla con todo lo que he puesto, adjunto su link (https://github.com/Juankkar/Programacion_Shell_Scripting_VIU/blob/main/Garcia_SnakeViral_2022_Nature/README.md).

Actividad 2 de la asignatura de Programación Shell Scripting

Máster en Bioinformática Valencia Internacional University (VIU)

Manipulación y formateo de archivos: Formato FASTQ y FASTA.

Credenciales del ator

- Autor: Juan Carlos García Estupiñán
- Año: 2022-2023
- Publicación: Nature

El propósito de esta actividad es desarrollar un flujo de trabajo completo (denominado en inglés pipeline) bioinformático que nos permita procesar una serie de datos biológicos. Con esta actividad, se pretende que el estudiante adquiera destrezas para interactuar con el sistema operativo a través de la línea de comandos y que sea capaz de desarrollar Shell scripts propios para resolver diferentes retos bioinformáticos focalizados en dos tipos de formato de texto, el formato FASTQ y en formato FASTA.

Directorios

En total hay 6 directorios:

analysis

- Directorio que contiene los análisis.

code

- scripts con el código usado usado: está formado por un total de 7 scripts para cada uno de los apartados.
 - [1nrow.sh](#): número de líneas de ambas réplicas de SRR1984406.
 - [2nsecuencias.sh](#): número de secuencias que presenta la réplica 1.
 - [3seq_identificar.sh](#): identificar el número de ocurrencias de la secuencia "ATGATG" en ambas réplicas.

~

Figura 5. Contenido del README en mi cuenta de Github.

Finalmente, genere una copia de esta plantilla cumplimentada y trasládela al directorio [submission/version1](#) donde se almacenará la primera versión de su trabajo realizado. **Para acabar este proyecto, vuelva a incluir una captura de pantalla con la estructura de directorios actual usando de nuevo el comando [tree](#) desde el terminal (0,25 pts).**

- Como mi intención es subir el contenido de este pipeline a mi cuenta de GitHub y sratoolkit es muy pesado, incluso después de comprimirlo con el siguiente comando `tar -zcvf sratoolkit.tar.gz sratoolkit.3.0.0-centos_linux64` (me daba ~83 Mb de tamaño por sí solo) lo he sustituido por un README con la cuenta de Github para descargar el programa. Igualmente adjunto un ls -l del escritorio donde he depositado sratoolkit para mostrar que lo he dejado fuera.

```
File Edit View Search Terminal Help
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo Desktop]$ ls -l
total 84384
drwxr-xr-x 5 UNIVERSIDADVIU\juan.carlos UNIVERSIDADVIU\domain users 4096 Nov 20 20:50 Programacion_Shell_Scripting_VIU
-rw-r--r-- 1 UNIVERSIDADVIU\juan.carlos UNIVERSIDADVIU\domain users 86400971 Nov 15 17:19 sratoolkit.tar.gz
drwxr-xr-x 4 UNIVERSIDADVIU\juan.carlos UNIVERSIDADVIU\domain users 4096 Sep 23 21:50 usr
[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo Desktop]$
```

- Los scripts con el código para los análisis se pueden ver en este enlace https://github.com/Juankkar/Programacion_Shell_Scripting_VIU/tree/main/Garcia_SnakeViral_2022_Nature/code.
- Finalmente adjunto el árbol final (En análisis tuve que añadir un README, no tiene nada en particular, pero si no añadía algo no me enviaba este directorio a mi repositorio con **git push**).

```
File Edit View Search Terminal Help

[UNIVERSIDADVIU\juan.carlos@a-3uv58hx3etnvo Garcia_SnakeViral_2022_Nature]$ tree
.
├── analysis
│   └── README.txt
├── code
│   ├── 1nrow.sh
│   ├── 2nsecuencias.sh
│   ├── 3seq_indentificar.sh
│   ├── 4fasta_format.sh
│   ├── 5randomseq.sh
│   ├── 6seq1.sh
│   └── 7dna_to_rna.sh
├── data
│   ├── processed
│   │   ├── all_sequences.fasta
│   │   ├── rna_secuencia.fasta
│   │   ├── secuencial.fasta
│   │   ├── secuencia2.fasta
│   │   ├── secuencia3.fasta
│   │   ├── secuencia4.fasta
│   │   └── secuencia5.fasta
│   └── raw
│       ├── SRR1984406_1.fastq
│       └── SRR1984406_2.fastq
├── LICENSE
├── README.md
├── submission
│   ├── version1
│   │   └── Actividad2_JuanCarlos.docx
│   └── version2
│       └── Actividad2_JuanCarlos.pdf
└── tools
    └── README.md

9 directories, 22 files
```

Referencias:

1. The Sequence Read Archive (SRA): Getting Started [Internet]. [citado 25 de noviembre de 2022]. Disponible en: <https://www.ncbi.nlm.nih.gov/sra/docs/>

2. Sequence File Formats | FASTQ & BCL formats for Illumina sequencing [Internet]. [citado 25 de noviembre de 2022]. Disponible en: <https://emea.illumina.com/informatics/sequencing-data-analysis/sequence-file-formats.html>
3. FASTQ files explained [Internet]. [citado 25 de noviembre de 2022]. Disponible en: <https://emea.support.illumina.com/bulletins/2016/04/fastq-files-explained.html>