**Linux**

**Pattern Matching (grep)**

Question 1: Which files contain the word “difference”?

Question 2 How many times does the word “Premium” occur in the file diamonds\_fix.txt?

Question 3 How many lines in the file diamonds\_fix.txt do not include the phrase “Very good”?

Question 1: grep -r difference   
1. ghandi.txt:The difference between what we do   
2. ghandi.txt.out:The difference   
3. ghandi\_2.txt:The difference between what we do   
4. ghandi\_cut.txt:The difference

Question 2: wc -w premium Diamonds\_fix.txt   
551492 Diamonds\_fix.txt   
551492 total

Question 3: grep -c -v "Very good" Diamonds\_fix.txt   
53941

**Bash Arrays**

Write a Bash script in which you:

Create an array called fruits which contains: pineapple, peach, raspberry, plum, apple and kiwi

Output the number of elements in the fruit array

Output the last element of the fruit array

#!/usr/bin/env bash

fruit=("pineapple" "peach" "raspberry" "plum" "apple" "kiwi")

echo "${#fruit[@]}"

echo "${fruit[5]}"

**Conditional Expressions**

Using the information in this and the previous section, write a Bash script called **temperature.sh** that:

reads in a command line argument into a variable called **temperature**

has a variable called **min\_temperature** and give it a variable of 10

has a variable called **max\_temperature** and give it a variable of 30

returns “Too hot!” if temperature is greater than **max\_temperature**

returns “Too cold!” if temperature is less than **min\_temperature**

returns “Just right!” if temperature is greater than the **min\_temperature** and less than **max\_temperature**

nano temperature.sh   
#!/usr/bin/env bash   
temperature=$1   
min\_temperature="10"   
max\_temperature="30"   
echo "temperature is: ${1}"   
echo "the lowest tolerable teperature is: ${min\_temperature}"   
echo "the highest tolerable teperature is: ${max\_temperature}"

if [[ ${temperature} -gt ${max\_temperature} ]] ; then   
echo "Too Hot"   
elif [[ ${temperature} -lt ${min\_temperature} ]] ; then   
echo "Too cold"   
elif [[ ${temperature} -gt ${min\_temperature} ]] && [[ ${min\_temperature} -lt ${max\_temperature} ]] ; then   
echo " the temperature is: just right"   
fi

**Switch Case Statements**

Create a Bash script called farm.sh that uses a case statement to perform the following functions:

Stores a command line argument in a variable called animal

Use a case switch statement which has the following conditions and responses

When the user enters cow, return “Here, moo”

When the user enters sheep, return “There a baa”

When the user enters duck, return “Everywhere a quack”

Otherwise, return “Old MacDonald had a farm”

#!/usr/bin/env bash   
#assign a variable animal   
animal=$1   
case $animal in   
cow)   
echo "Here moo"   
;;   
sheep)   
echo "There baa"   
;;   
duck)   
echo "Everywhere quack"   
;;   
\*)   
echo "Old MacDonald has a farm"   
;;   
esac

**Exercise and discuss**

For this exercise, what we’d like you to do is create an interactive Bash script called diamonds.sh.

First, prompt the user to provide a cut and store their response in a variable called quality.

If the user enters either Fair or Good, return Insufficient quality to proceed.

If the user enters a value other than Premium, Ideal or Very Good, return Invalid cut.

Otherwise, count the number of diamonds (rows) which have the user-defined cut. Hint: look at the usage for grep to find an option that will count the number of lines which match a given pattern.

#!/usr/bin/env bash

read -p "Enter your cut " quality

if [[ ${quality} == "Fair" ]] || [[ ${quality} == "Good" ]]

then

echo "Insufficient quality to proceed"

elif [[ ${quality} != "Premium" ]] && [[ ${quality} != "Ideal" ]] && [[ ${quality} != "VeryGood" ]]

then

echo "Invalid cut"

else

grep -c "${quality}" Diamonds.csv

fi

**For Loops**

Create a for loop which iterates from 1 to 5 in increments of 1. If the value is 2 return “fizz” otherwise, return “buzz”

#!/usr/bin/env bash   
for ((n=1; n<=5; n++))   
do   
echo "integer: ${n}"   
done   
read -p "enter an integer: " n

if [[ $n -eq 2 ]] ; then   
echo "fizz"   
else   
echo "buzz"   
fi

**Bash Functions**

Create a function called file\_exists taking the first argument (a filename) which it uses to see if the file exists. If it doesn’t, return “File does not exist: “, followed by the filename.

function file\_exist() { read -p "enter the file name: " txt   
if [[ -e $txt ]] ; then   
echo "file exist"   
else   
echo "file does not exist"   
fi   
}   
file\_exist   
file\_exist   
*# another way to write the code ignoring interactiveness*   
function file\_exist2() { if [[ ! -e ${1} ]] ; then   
echo "file does not exist"   
else   
echo "file exist"   
fi   
}   
file\_exist2 me.txt   
file\_exist2 thin.txt

RESULTS   
enter the file name: me.txt   
file exist   
enter the file name: hyg   
file does not exist

#for the second code   
file exist   
file does not exist

**File Manipulation: sort and uniq**

How would you extract only the lines that repeat more than once in the file fruit.txt into a new file named repeated\_fruit.txt file?

sort fruit.txt | uniq -id > repeated\_fruit.txt

**Final Exercise on Bash scripting - Use Bash Scripting to Parse Biological Data**

sample\_10000\_11000.bam, sample\_11000\_12000.bam, and sample\_12000\_13000.bam are three data files.

Task:

Using a Bash script, get the number of records for each of the three example data files.

Some hints:

Use comments

Use the set command

Check whether each file is empty before running samtools

Use a loop – i.e. don’t run three samtools commands with hardcoded filenames, use wildcards (e.g. sample\*.bam where \* matches any string)

Return the filename and the number of records back to the user

#!/usr/bin/env bash

set -eu

#print the name of file   
for file in \*.bam   
do   
echo "filename: ${file}"

#check if the file are empty   
if [[ -s ${file} ]] ; then   
echo "is not empty"   
fi

# Count the number of records using samtools view and set the output as a variable   
alignments=$(samtools view -c "${file}")

# Return the filename and the number of records back to the user   
echo "And the number of alignments in ${file} is ${alignments}"

done

RESULT   
filename: sample\_10000\_11000.bam   
is not empty   
And the number of alignments in sample\_10000\_11000.bam is 1947   
filename: sample\_11000\_12000.bam   
is not empty   
And the number of alignments in sample\_11000\_12000.bam is 123   
filename: sample\_12000\_13000.bam   
is not empty   
And the number of alignments in sample\_12000\_13000.bam is 276

**R-programming**

**Creating and Manipulating Variables and Vectors in R**

Question 1. Create 3 vectors:

a vector x containing the numbers 3, 10 and 30

a vector m containing the content of x repeated twice

a vector n containing two copies of x separated by a 0

Question 2. Is the content of m equal to the content of n?

Question 3. Note that you should also obtain a warning message because the 2 vectors are not of the same length. How can you check the length of both vectors?

> x = c(3, 10, 30)   
> x   
[1] 3 10 30   
> m = rep(x, times=2)   
> m   
[1] 3 10 30 3 10 30   
> n = c(x, 0, x)   
> n   
[1] 3 10 30 0 3 10 30   
> m==n   
[1] TRUE TRUE TRUE FALSE FALSE FALSE FALSE   
Warning message:   
In m == n : longer object length is not a multiple of shorter object length   
> length(m)   
[1] 6   
> length(n)   
[1] 7

**Working on Data Frames - Exercises and Discussion**

Question 1. Create a Data frame

Create a Data frame called df\_fruits from the following Vectors:

A Vector called Fruits composed of Apple, Banana, Orange, Mango

A Vector called Price composed of 4, 3, 2, 8

A Vector called Nature composed of Local, Exotic, Local, Exotic as factors

Question 2. Check your Data frame

How would you verify that the newly Data frame called df\_fruits

Is properly created

What are the levels of the factors in the column called Nature

How many different of these levels were created

Question 3. Manipulate your Data frame

How would you verify the number of rows and columns of the new Data frame called df\_fruits?

What class of data is in each column?

How would you select all elements of the column called Price?

Question 1   
> fruit = c("Apple", "Banana", "Orange", "Mango")   
> price = c(4,3,2,8)   
> nature = as.factor( c("Local", "Exotic", "Local", "Exotic"))   
> df\_fruit = data.frame ( row.names = fruits, price, nature)

Question 2   
a).   
> df\_fruit   
price nature   
Apple 4 Local   
Banana 3 Exotic   
Orange 2 Local   
Mango 8 Exotic   
>str(df\_fruit)   
'data.frame': 4 obs. of 2 variables:   
$ price : num 4 3 2 8   
$ nature: Factor w/ 2 levels "Exotic","Local": 2 1 2 1

b).   
> levels(nature)   
[1] "Exotic" "Local"

c).   
nlevels(nature)   
[1] 2

Question 3   
1.   
> nrow(df\_fruit)   
[1] 4   
> ncol(df\_fruit)   
[1] 2

2.   
> sapply(df\_fruit, class)   
price nature   
"numeric" "factor"

3.   
> df\_fruit[[1]]   
[1] 4 3 2 8 OR   
> df\_fruit[1]   
price   
Apple 4   
Banana 3   
Orange 2   
Mango 8