BINF2111 – Introduction to Bioinformatics Computing

BASH 101 – while wild loops of function

Richard Allen White III, PhD
RAW Lab

Lecture 11 – Thursday Sep 30th, 2021

Learning Objectives

- Review quiz/bonus

- Review lab 4

- Review bash for loops

- Bash while loops

- Bash functions

- Quiz 11

Quiz 10 answers

```
My input is:              My output is:
more file.tsv             more file.csv
bill rod david             bill,rod,david
Xi abdul larry             Xi,abdul,larry
```

```
perl -pi -e 's/\t/,/' file.tsv
```

T or F

Quiz 10 answers

My input is:

```
more file.tsv
bill rod david
Xi abdul larry
```

My output is:

```
more file.csv
bill,rod david
Xi,abdul larry
```

```
perl -pi -e 's/\t/,/' file.tsv
```

T or F

70 % missed this one.

Quiz 10 answers

My input is:

```
more file.tsv
bill rod david
Xi abdul larry
```

command: `head -1 file.txt | tail +2`

What will the command do?

Quiz 10 answers

My input is:

more file.tsv

bill rod david

Xi abdul larry

command: head -1 file.txt | tail +2

What will the command do?

Will not work due to errors (40% missed)

WHY?

Quiz 10 answers

My input is:

```
more file.tsv
bill rod david
Xi abdul larry
bill rod david
Xi abdul larry
steve bill larry
```

What is the command to print only lines 2 and 5 with the correct line numbers?

Quiz 10 answers

My input is:

```
more file.tsv
bill rod david
Xi abdul larry
bill rod david
Xi abdul larry
steve bill larry
```

What is the command to print only lines 2 **and** 5 with the correct line numbers? (50% missed)

```
cat --number file.tsv | sed -n '2p;5p'
```

OR

```
cat -n file.tsv | sed -n '2p;5p'
```

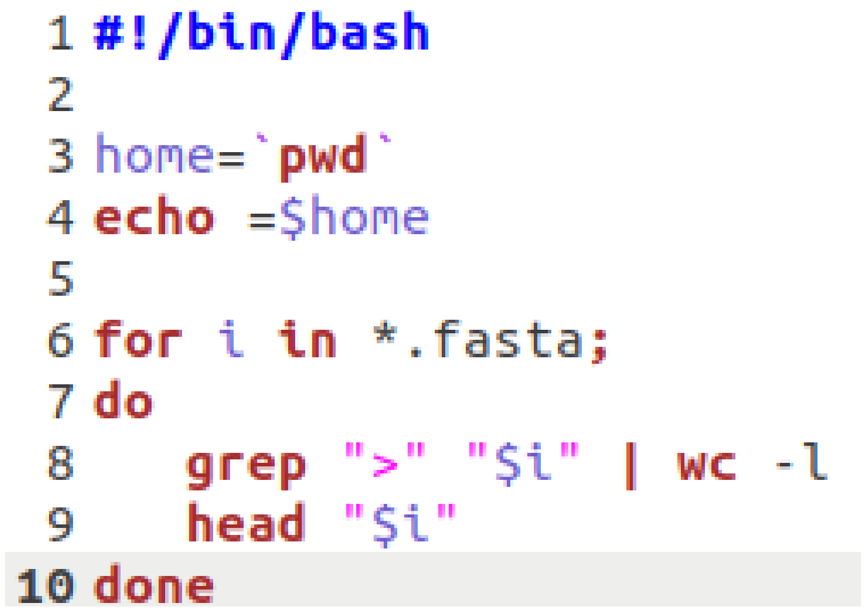
(Mac/Linux)

Bonus 9

- Write a bash script that prints the working directory, counts all the sequences within a fasta files within the working directory, and prints the first five lines of the file into std_out.txt?

Bonus 9

- Write a bash script that prints the working directory, counts all the sequences within a fasta files within the working directory, and prints the first five lines of the file into std_out.txt?



```
#!/bin/bash

home=`pwd`
echo =$home

for i in *.fasta;
do
    grep ">" "$i" | wc -l
    head "$i"
done
```

Lab Question 3

Write a bash script to count the number of ATG (starts) and TAA, TAG, TGA (stops) from the example2.fasta file.

Remember that ATG encodes for methionine so the only count the from the beginning of the sequence or the end for the stops.

HOW WOULD YOU DO THIS?

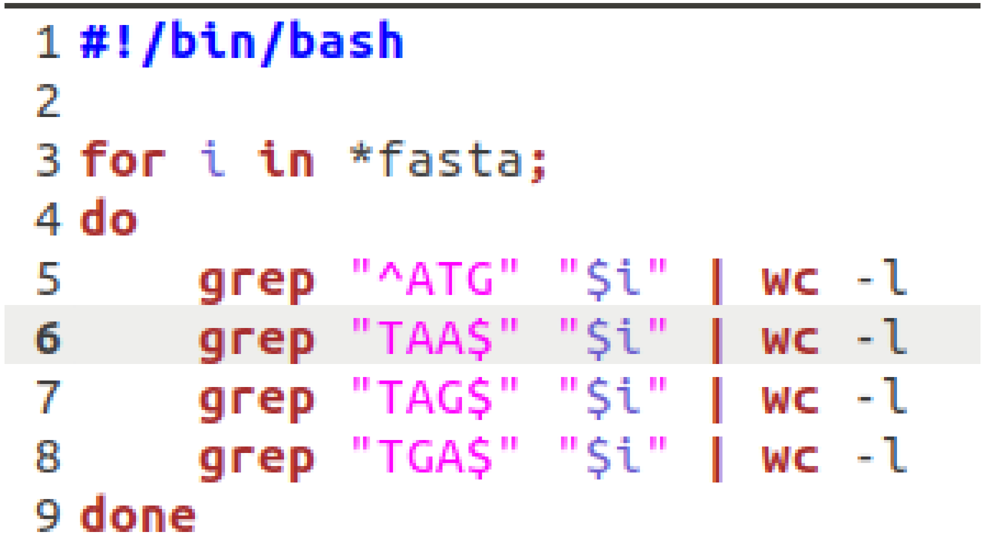
Lab Question 3

Write a bash script to count the number of ATG (starts) and TAA, TAG, TGA (stops) from the example2.fasta file.

Remember that ATG encodes for methionine so the only count the from the beginning of the sequence or the end for the stops.

HOW WOULD YOU DO THIS?

BETTER WAY?



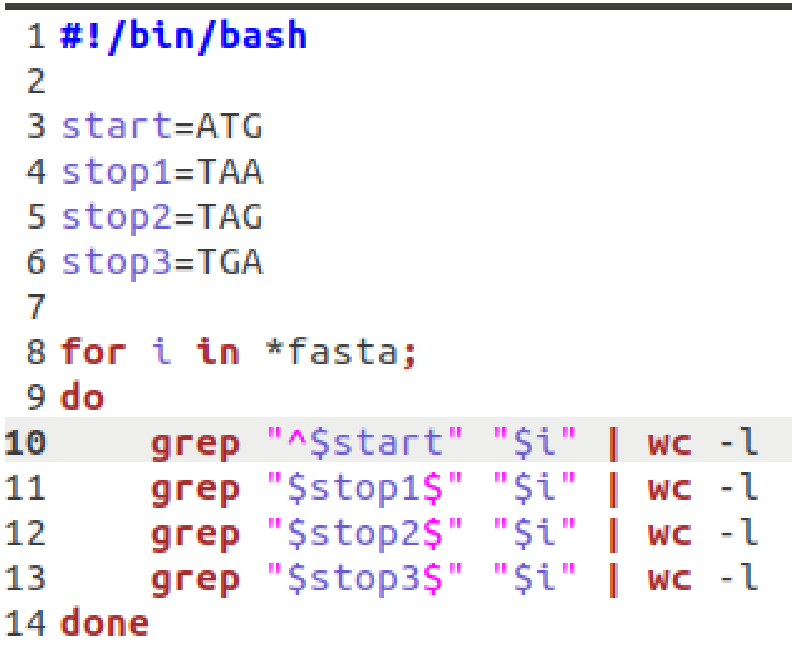
```
#!/bin/bash

for i in *fasta;
do
      grep "^ATG" "$i" | wc -l
      grep "TAA$" "$i" | wc -l
      grep "TAG$" "$i" | wc -l
      grep "TGA$" "$i" | wc -l
done
```

Lab Question 3

Write a bash script to count the number of ATG (starts) and TAA, TAG, TGA (stops) from the example2.fasta file.

Remember that ATG encodes for methionine so the only count the from the beginning of the sequence or the end for the stops.

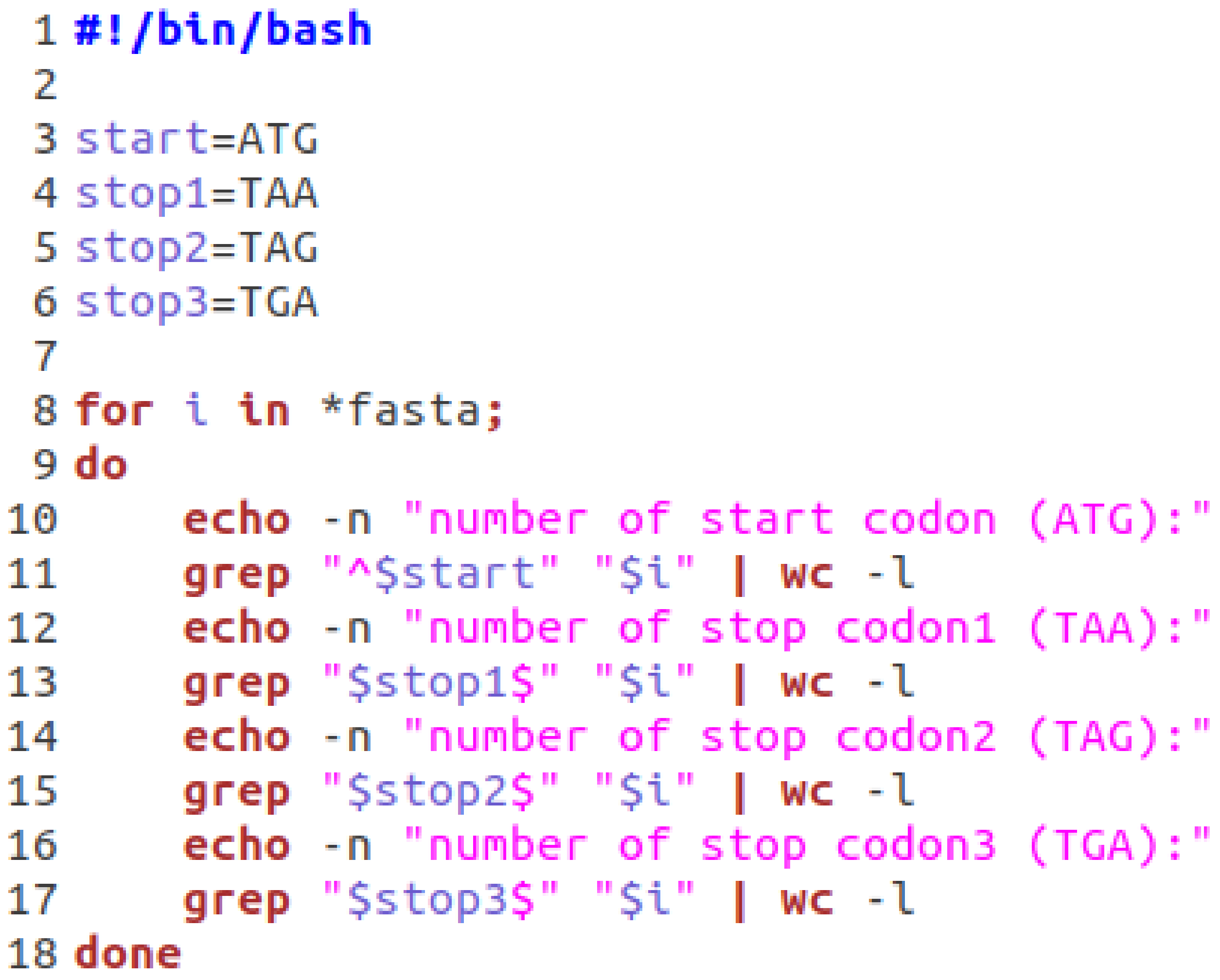
EVEN BETTER?



```bash
#!/bin/bash

start=ATG
stop1=TAA
stop2=TAG
stop3=TGA

for i in *fasta;
do
    grep "^$start" "$i" | wc -l
    grep "$stop1$" "$i" | wc -l
    grep "$stop2$" "$i" | wc -l
    grep "$stop3$" "$i" | wc -l
done
```

Lab Question 3

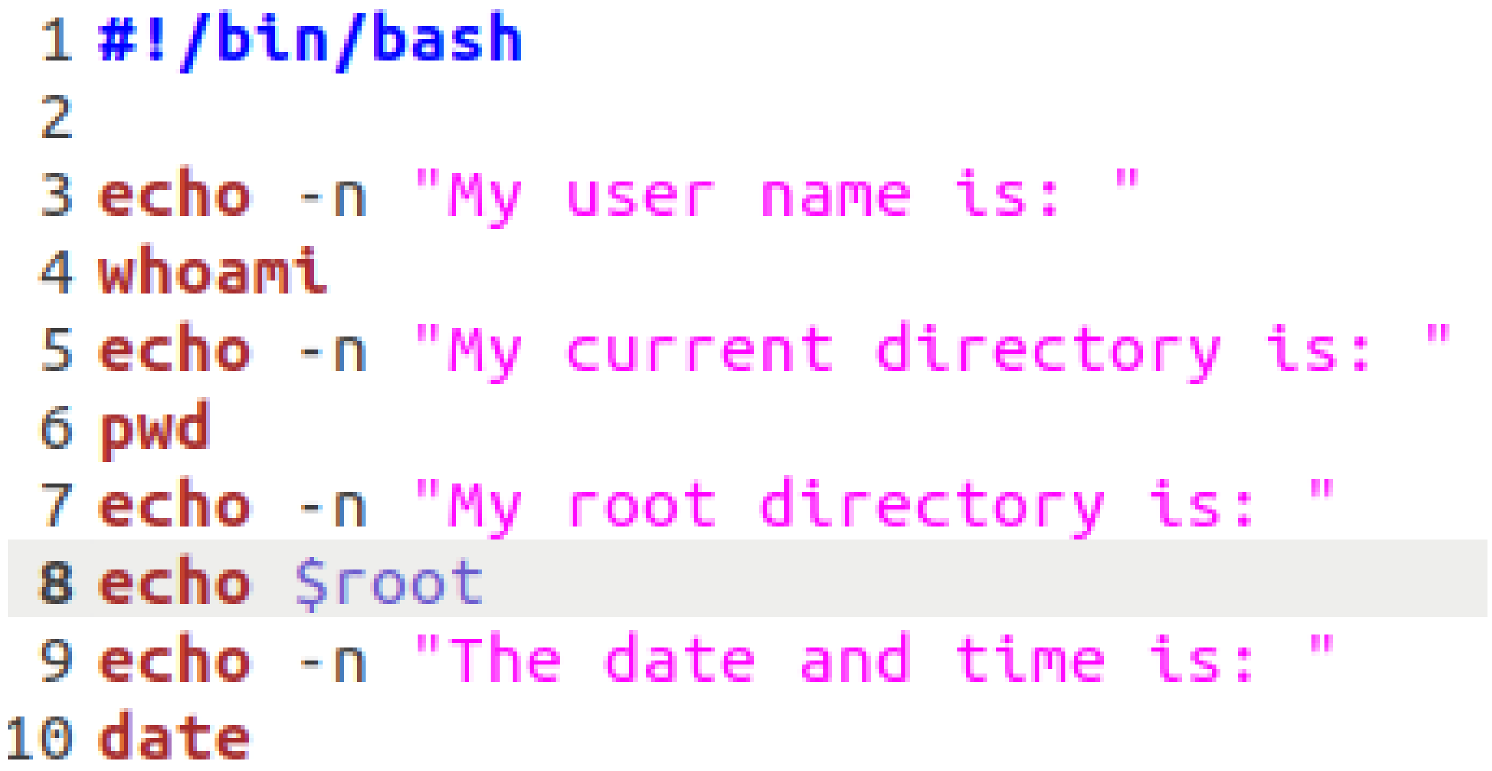
EVEN BETTER?


```
#!/bin/bash

start=ATG
stop1=TAA
stop2=TAG
stop3=TGA

for i in *fasta;
do
	echo -n "number of start codon (ATG):"
	grep "^$start" "$i" | wc -l
	echo -n "number of stop codon1 (TAA):"
	grep "$stop1$" "$i" | wc -l
	echo -n "number of stop codon2 (TAG):"
	grep "$stop2$" "$i" | wc -l
	echo -n "number of stop codon3 (TGA):"
	grep "$stop3$" "$i" | wc -l
done
```

Lab Question 4

Write a bash script that tells me my username, current directory, the location of my root directory, and the date/time

HOW WOULD YOU DO THIS?



```bash
#!/bin/bash

echo -n "My user name is: "
whoami
echo -n "My current directory is: "
pwd
echo -n "My root directory is: "
echo $root
echo -n "The date and time is: "
date
```

Lab Question 4

Write a bash script that tells me my username, current directory, the location of my root directory, and the date/time

HOW WOULD YOU DO THIS?

```
bash script_date.sh
```

```
My user name is: docwhite
My current directory is: /home/docwhite/Desktop
My root directory is:
The date and time is: Tue Sep 28 19:40:29 EDT 2021
```

BASH - for loop

```
for i in file.*;do
    command $i
done
```

BASH - for loop (C-style)

```
for ((i = 0 ; i < 100 ; i++)); do
  command $i
done
```

BASH - for loop (Java)

```
for (i = 0 ; i < 100 ; i++){
    command (i);
}
```

BASH - for loop (Python)

```
for x in file:
    command (x)
```

BASH - while loop

```
while[ condition ]
do
   command1
   command2
   command3
done
```

BASH - while loop

Command1 to Command3 will be executed repeatedly till condition is false. The argument for a while loop can be any boolean expression. Infinite loops occur when the conditional never evaluates to false. The while loop should be used as long as a certain condition is true, such as the a counter is less than a maximum value or the ping time to a server is lower than a threshold or forever if you loop while TRUE or while 1.

Here is the while loop one-liner syntax:

while [condition]; **do** commands; **done**

while control-command; **do** COMMANDS; **done**

BASH - while loop

```
#!/bin/bash
x=1
while [ $x -le 5 ]
do
  echo "Welcome $x times"
  x=$(( $x + 1 ))
done
```

BASH - while loop (one - liner)

```
x=1; while [ $x -le 5 ]; do echo "Welcome $x times" $(( x++ )); done
```

BASH - while loop (read line by line)

```
#!/bin/bash
FILE=$1
# read $FILE using the file descriptors
exec 3<&0
exec 0<$FILE
while read line
do
    # use $line variable to process line
    echo $line
done
exec 0<&3
```

BASH - until loop

The until loop is similar to the while loop but with reverse logic. Instead of looping while a condition is true you are assuming the condition is false and looping until it becomes true. They are reverse of each other in logical expression.

```
until [ CONDITION ]; do
  LINES OF CODE
  MORE LINES OF CODE
done
```

BASH - until loop

```
#!/bin/bash
NUM=1
Until [ “$NUM” -gt 1000 }; do
    echo $NUM
    let NUM=NUM*2
done
```

BASH - functions

```
Function_name( ){
  command
}
```

Think of a function as a small script within a script. It's a small chunk of code which you may call multiple times within your script.

MY FAVORITE WAY! (There is another way)

BASH - functions

```
Function function_name( ){
    command
}
```

Not my favorite. But, you may like it?

BASH - functions

```
Function function_name( ){
    command
}
```

Not my favorite. But, you may like it?

BASH – Passing Arguments/Return values

```
#!/bin/bash


print_this() {
  echo Hello $1
    return 5
}
print_this Mars
echo The previous function has a return value of $?
```

Quiz 11

- On canvas now

Bonus 10

- Write a function that will return the number of lines it has in it?