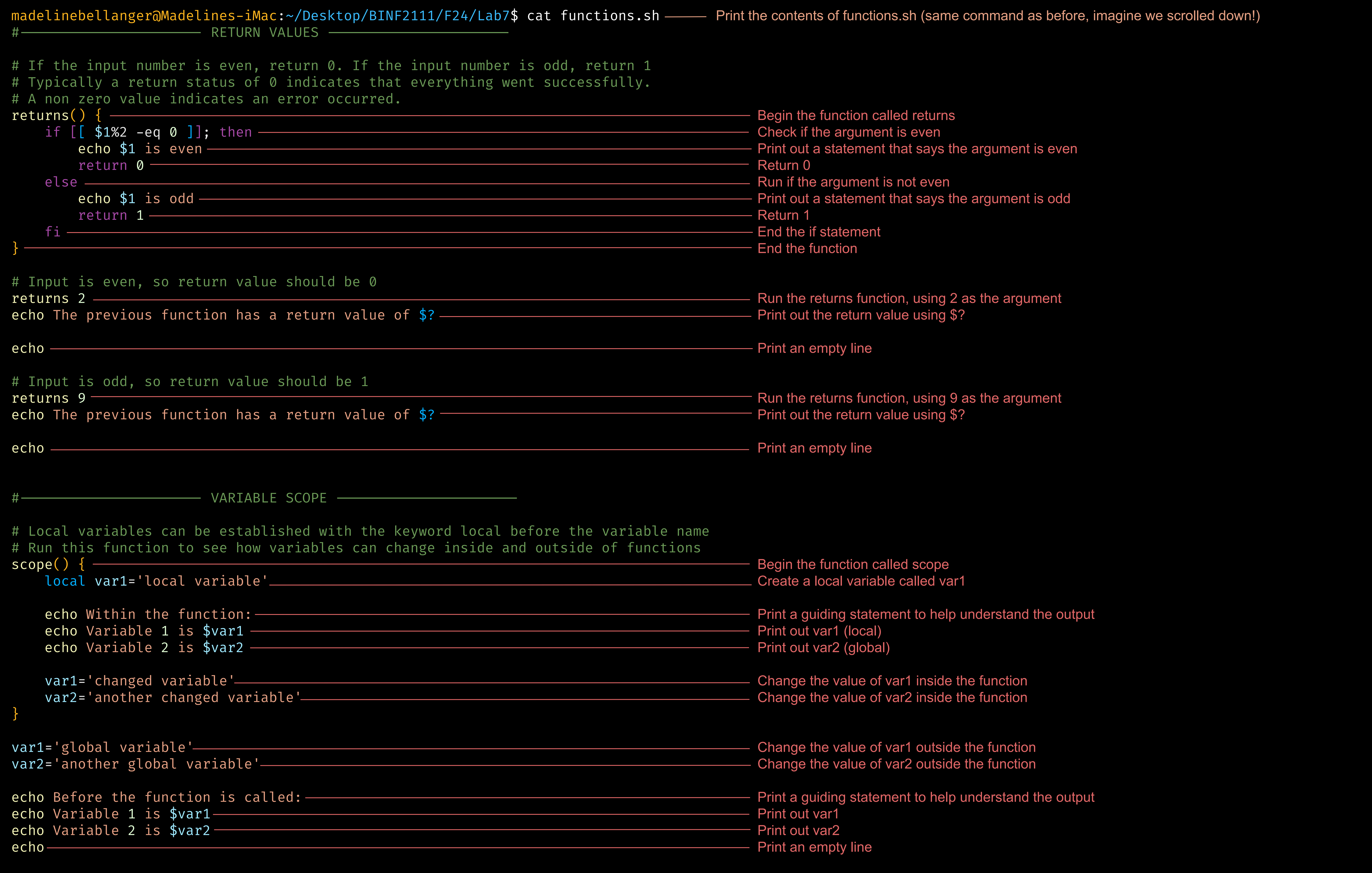
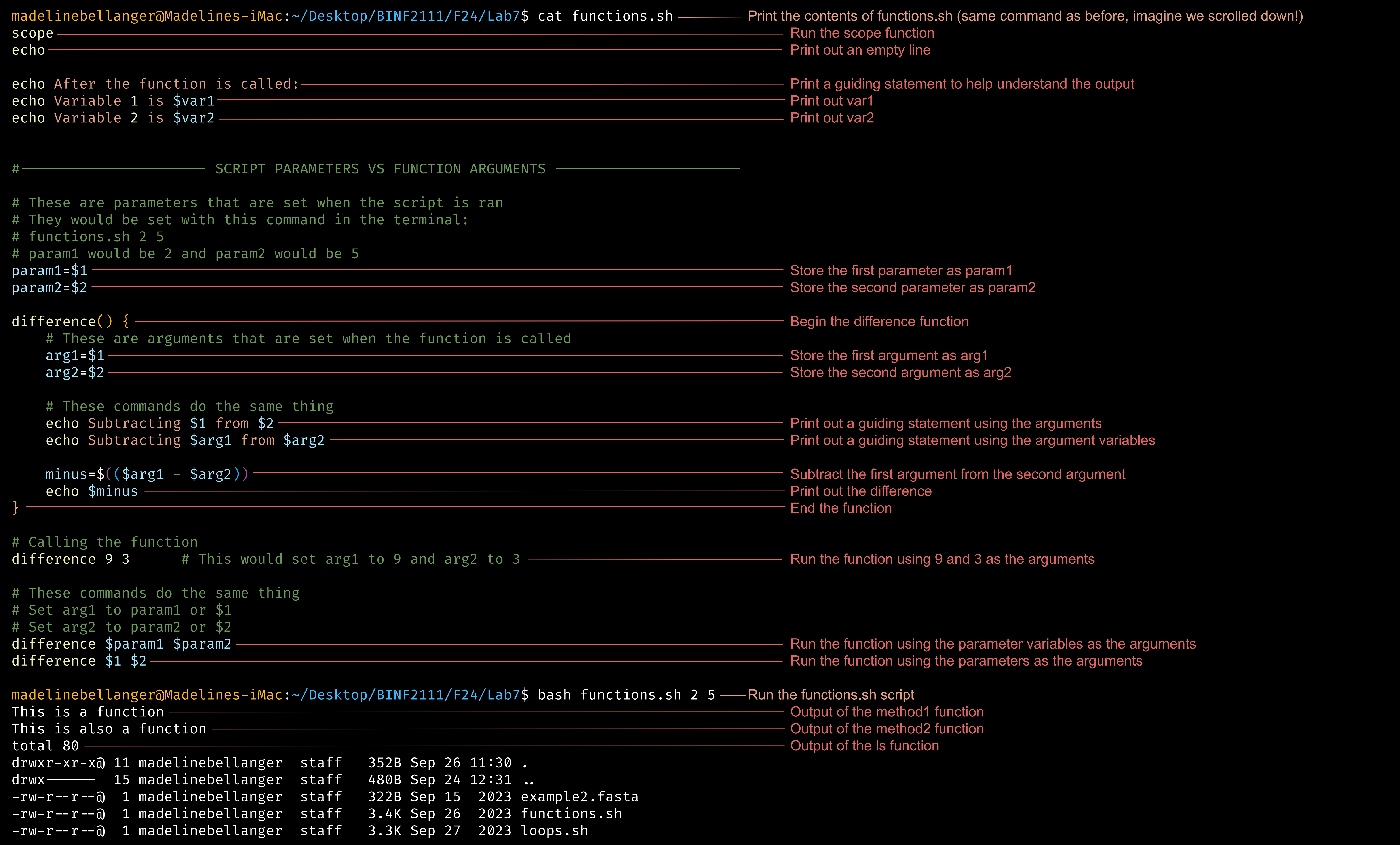
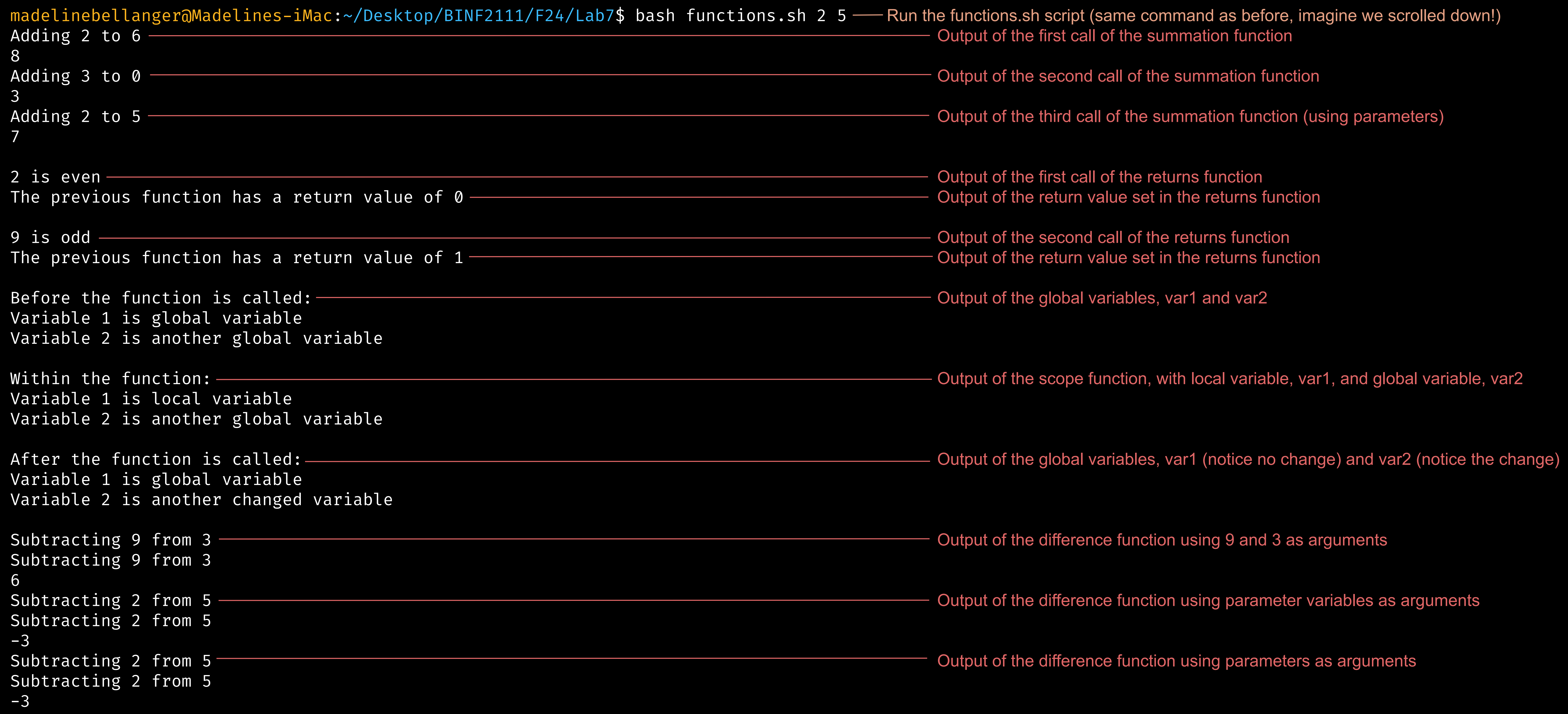
```
madelinebellanger@Madelines-iMac:~/Desktop/BINF2111/F24/Lab7$ cat functions.sh ———— Print the contents of functions.sh
#!/bin/bash ———————————————————————————————————————————— Include the shebang in the script!

#—————————————————— BASIC FUNCTIONS ——————————————————

# Method #1: Creating a function
#            Note the parenthesis after the function's name
method1() { ———————————————————————————————————————————— Begin the first function called method1
    echo This is a function ———————————————————————————— Print out a statement in the function
} —————————————————————————————————————————————————————— End the method1 function

# Method #2: Creating a function
#            Note the lack of parenthesis and the word function
function method2 { ————————————————————————————————————— Begin the second function called method2
    echo This is also a function ——————————————————————— Print out a statement in the function
} —————————————————————————————————————————————————————— End the method2 function

# Calling both functions
method1 ———————————————————————————————————————————————— Run the method1 function
method2 ———————————————————————————————————————————————— Run the method2 function

# Overriding commands: Make a function that goes by the same name as a preset command,
#                      but does an enhanced version of that command.
#                      The word command before ls is essential to this working!
ls() { ————————————————————————————————————————————————— Begin the function called ls
    command ls -alh ———————————————————————————————————— Change the functionality of the ls command so that is runs ls -alh instead of ls
} —————————————————————————————————————————————————————— End the ls function

# Calling the function
ls ————————————————————————————————————————————————————— Run the ls function

echo ——————————————————————————————————————————————————— Print out an empty line


#—————————————————— ARGUMENTS ——————————————————

# Use $1 and $2 to set arguments in the function that prints 2 numbers, adds them
# together, and prints the sum
summation() { —————————————————————————————————————————— Begin the summation function
    echo Adding $1 to $2 ——————————————————————————————— Print out a statement that takes in two arguments
    sum=$(($1 + $2)) ——————————————————————————————————— Add the two arguments together and set it equal to sum
    echo $sum —————————————————————————————————————————— Print out the sum
}

summation 2 6   # will add 2+6 ————————————————————————— Run the summation function using 2 and 6
summation 3 0   # will add 3+0 ————————————————————————— Run the summation function using 3 and 0
summation $1 $2 # will take in two numbers given as parameters when the script is ran ——— Run the summation function using values taken in from running the script
                # and insert them into the function's arguments

echo ——————————————————————————————————————————————————— Print out an empty line
```



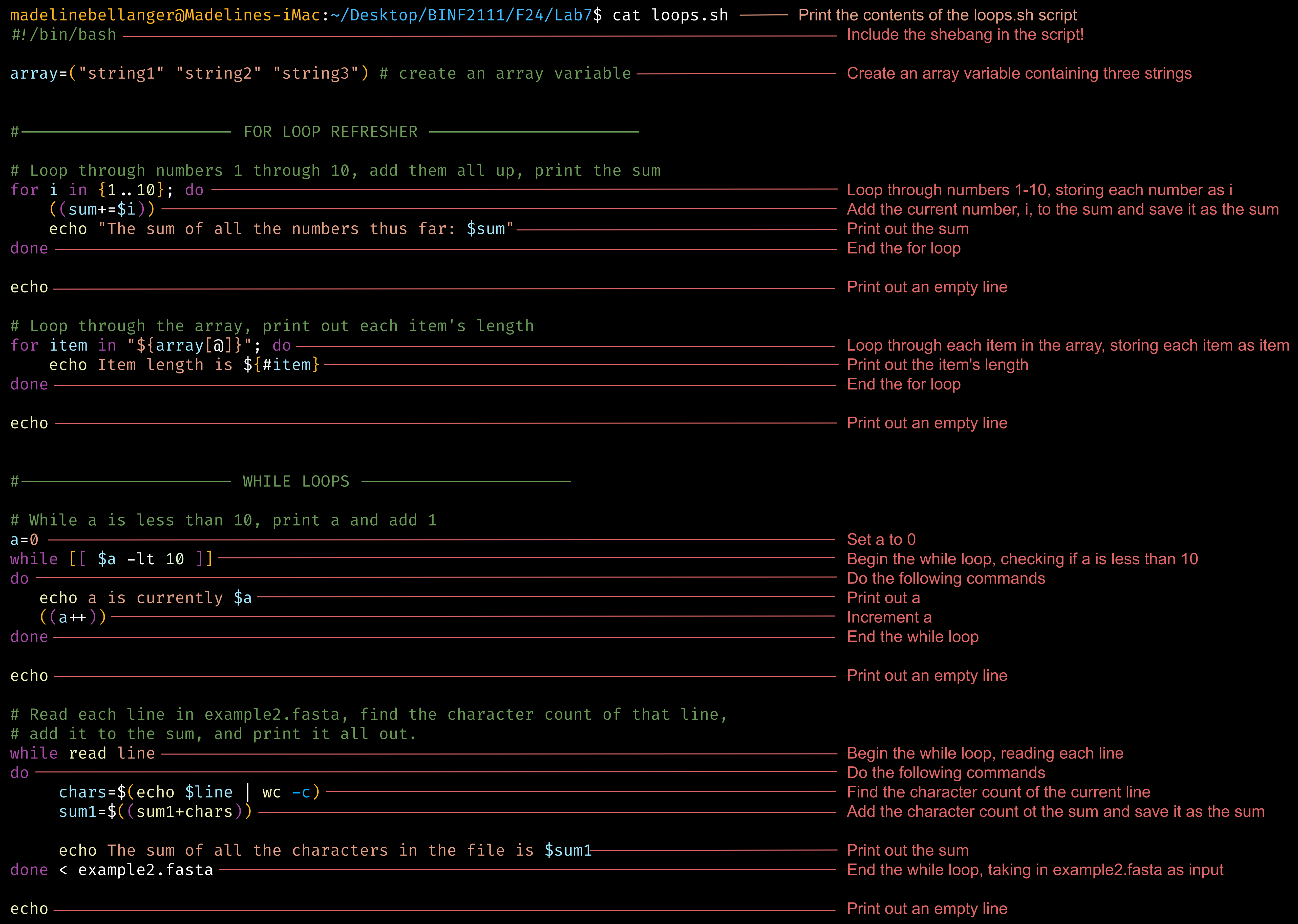
```
madelinebellanger@Madelines-iMac:~/Desktop/BINF2111/F24/Lab7$ cat functions.sh ——— Print the contents of functions.sh (same command as before, imagine we scrolled down!)
#——————————————— RETURN VALUES ———————————————

# If the input number is even, return 0. If the input number is odd, return 1
# Typically a return status of 0 indicates that everything went successfully.
# A non zero value indicates an error occurred.
returns() { ——————————————————————————— Begin the function called returns
    if [[ $1%2 -eq 0 ]]; then ————————————— Check if the argument is even
        echo $1 is even ———————————————— Print out a statement that says the argument is even
        return 0 ——————————————————— Return 0
    else ——————————————————————————— Run if the argument is not even
        echo $1 is odd ————————————————— Print out a statement that says the argument is odd
        return 1 ——————————————————— Return 1
    fi ————————————————————————————— End the if statement
} ———————————————————————————————— End the function

# Input is even, so return value should be 0
returns 2 ——————————————————————————— Run the returns function, using 2 as the argument
echo The previous function has a return value of $? ——— Print out the return value using $?

echo ———————————————————————————————— Print an empty line

# Input is odd, so return value should be 1
returns 9 ——————————————————————————— Run the returns function, using 9 as the argument
echo The previous function has a return value of $? ——— Print out the return value using $?

echo ———————————————————————————————— Print an empty line


#——————————————— VARIABLE SCOPE ———————————————

# Local variables can be established with the keyword local before the variable name
# Run this function to see how variables can change inside and outside of functions
scope() { ——————————————————————————— Begin the function called scope
    local var1='local variable'————————————— Create a local variable called var1

    echo Within the function: ———————————— Print a guiding statement to help understand the output
    echo Variable 1 is $var1 ————————————— Print out var1 (local)
    echo Variable 2 is $var2 ————————————— Print out var2 (global)

    var1='changed variable'————————————— Change the value of var1 inside the function
    var2='another changed variable'————————— Change the value of var2 inside the function
}

var1='global variable'——————————————————— Change the value of var1 outside the function
var2='another global variable'—————————————— Change the value of var2 outside the function

echo Before the function is called: ———————— Print a guiding statement to help understand the output
echo Variable 1 is $var1———————————————— Print out var1
echo Variable 2 is $var2———————————————— Print out var2
echo———————————————————————————————— Print an empty line
```



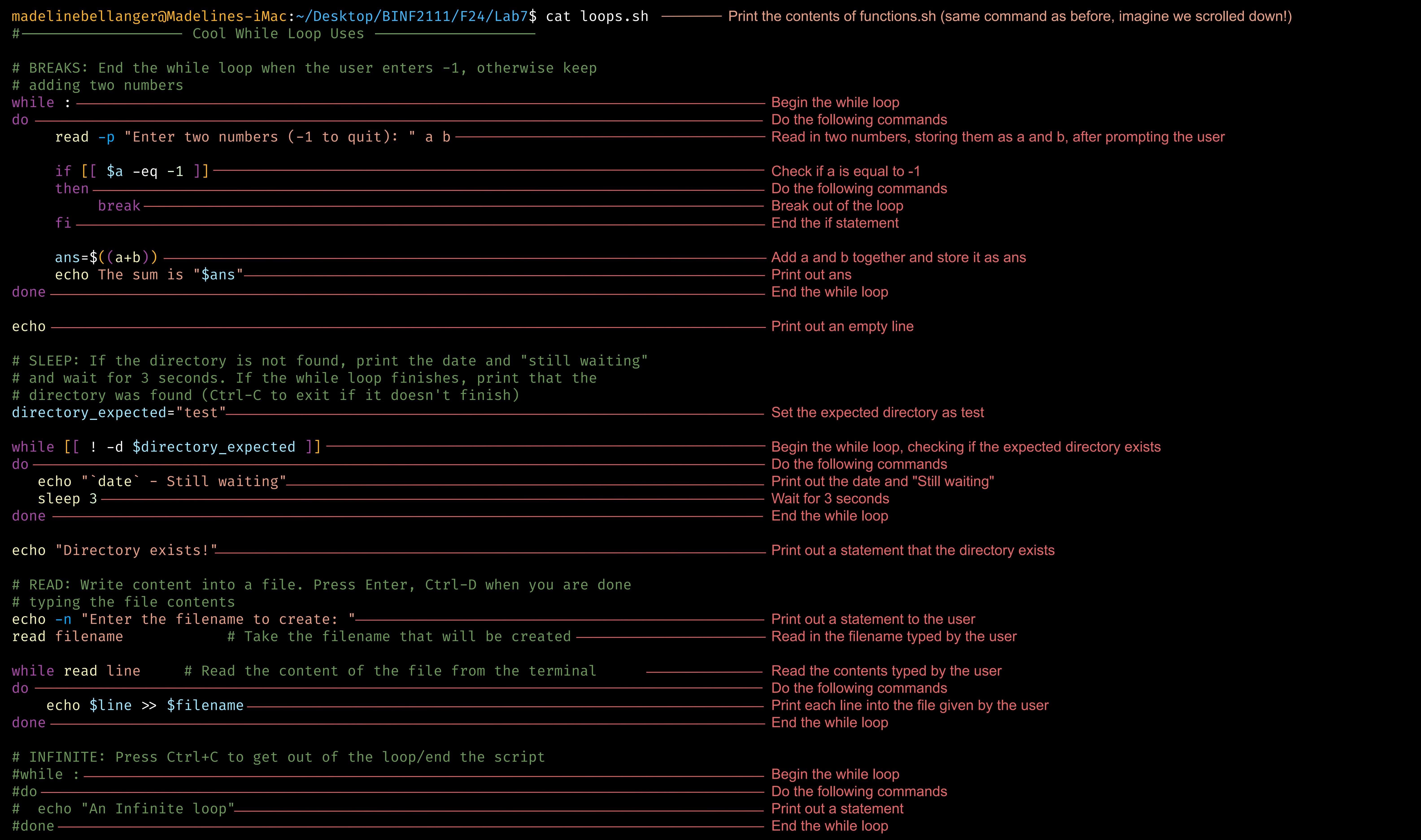
```
madelinebellanger@Madelines-iMac:~/Desktop/BINF2111/F24/Lab7$ cat functions.sh ——— Print the contents of functions.sh (same command as before, imagine we scrolled down!)
scope ——— Run the scope function
echo ——— Print out an empty line

echo After the function is called: ——— Print a guiding statement to help understand the output
echo Variable 1 is $var1 ——— Print out var1
echo Variable 2 is $var2 ——— Print out var2


#———————————————— SCRIPT PARAMETERS VS FUNCTION ARGUMENTS ————————————————

# These are parameters that are set when the script is ran
# They would be set with this command in the terminal:
# functions.sh 2 5
# param1 would be 2 and param2 would be 5
param1=$1 ——— Store the first parameter as param1
param2=$2 ——— Store the second parameter as param2

difference() { ——— Begin the difference function
    # These are arguments that are set when the function is called
    arg1=$1 ——— Store the first argument as arg1
    arg2=$2 ——— Store the second argument as arg2

    # These commands do the same thing
    echo Subtracting $1 from $2 ——— Print out a guiding statement using the arguments
    echo Subtracting $arg1 from $arg2 ——— Print out a guiding statement using the argument variables

    minus=$(($arg1 - $arg2)) ——— Subtract the first argument from the second argument
    echo $minus ——— Print out the difference
} ——— End the function

# Calling the function
difference 9 3      # This would set arg1 to 9 and arg2 to 3 ——— Run the function using 9 and 3 as the arguments

# These commands do the same thing
# Set arg1 to param1 or $1
# Set arg2 to param2 or $2
difference $param1 $param2 ——— Run the function using the parameter variables as the arguments
difference $1 $2 ——— Run the function using the parameters as the arguments

madelinebellanger@Madelines-iMac:~/Desktop/BINF2111/F24/Lab7$ bash functions.sh 2 5 ——— Run the functions.sh script
This is a function ——— Output of the method1 function
This is also a function ——— Output of the method2 function
total 80 ——— Output of the ls function
drwxr-xr-x@ 11 madelinebellanger  staff   352B Sep 26 11:30 .
drwx------  15 madelinebellanger  staff   480B Sep 24 12:31 ..
-rw-r--r--@  1 madelinebellanger  staff   322B Sep 15  2023 example2.fasta
-rw-r--r--@  1 madelinebellanger  staff   3.4K Sep 26  2023 functions.sh
-rw-r--r--@  1 madelinebellanger  staff   3.3K Sep 27  2023 loops.sh
```



```
madelinebellanger@Madelines-iMac:~/Desktop/BINF2111/F24/Lab7$ bash functions.sh 2 5 —— Run the functions.sh script (same command as before, imagine we scrolled down!)
Adding 2 to 6 ————————————————————————— Output of the first call of the summation function
8
Adding 3 to 0 ————————————————————————— Output of the second call of the summation function
3
Adding 2 to 5 ————————————————————————— Output of the third call of the summation function (using parameters)
7

2 is even ————————————————————————————— Output of the first call of the returns function
The previous function has a return value of 0 —— Output of the return value set in the returns function

9 is odd —————————————————————————————— Output of the second call of the returns function
The previous function has a return value of 1 —— Output of the return value set in the returns function

Before the function is called: ———————— Output of the global variables, var1 and var2
Variable 1 is global variable
Variable 2 is another global variable

Within the function: —————————————————— Output of the scope function, with local variable, var1, and global variable, var2
Variable 1 is local variable
Variable 2 is another global variable

After the function is called: ————————— Output of the global variables, var1 (notice no change) and var2 (notice the change)
Variable 1 is global variable
Variable 2 is another changed variable

Subtracting 9 from 3 —————————————————— Output of the difference function using 9 and 3 as arguments
Subtracting 9 from 3
6
Subtracting 2 from 5 —————————————————— Output of the difference function using parameter variables as arguments
Subtracting 2 from 5
-3
Subtracting 2 from 5 —————————————————— Output of the difference function using parameters as arguments
Subtracting 2 from 5
-3
```

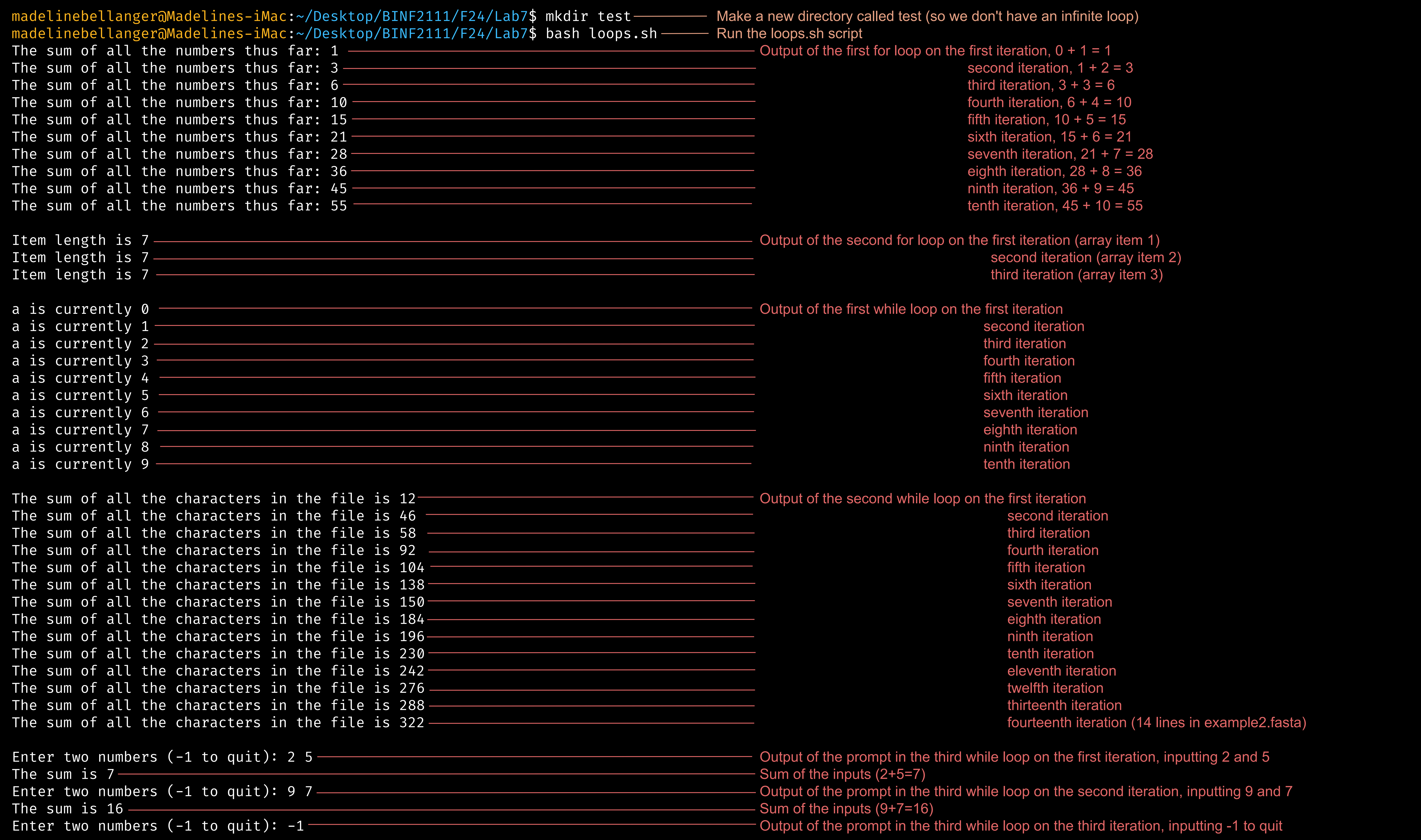


```
madelinebellanger@Madelines-iMac:~/Desktop/BINF2111/F24/Lab7$ cat loops.sh ——— Print the contents of the loops.sh script
#!/bin/bash ———————————————————————————————————————————————————————————— Include the shebang in the script!

array=("string1" "string2" "string3") # create an array variable ——————— Create an array variable containing three strings


#——————————————————— FOR LOOP REFRESHER ———————————————————

# Loop through numbers 1 through 10, add them all up, print the sum
for i in {1..10}; do ———————————————————————————————————————————————————— Loop through numbers 1-10, storing each number as i
    ((sum+=$i)) ————————————————————————————————————————————————————————— Add the current number, i, to the sum and save it as the sum
    echo "The sum of all the numbers thus far: $sum"————————————————————— Print out the sum
done ———————————————————————————————————————————————————————————————————— End the for loop

echo ———————————————————————————————————————————————————————————————————— Print out an empty line

# Loop through the array, print out each item's length
for item in "${array[@]}"; do ——————————————————————————————————————————— Loop through each item in the array, storing each item as item
    echo Item length is ${#item} ———————————————————————————————————————— Print out the item's length
done ———————————————————————————————————————————————————————————————————— End the for loop

echo ———————————————————————————————————————————————————————————————————— Print out an empty line


#——————————————————— WHILE LOOPS ———————————————————

# While a is less than 10, print a and add 1
a=0 ————————————————————————————————————————————————————————————————————— Set a to 0
while [[ $a -lt 10 ]] ——————————————————————————————————————————————————— Begin the while loop, checking if a is less than 10
do —————————————————————————————————————————————————————————————————————— Do the following commands
   echo a is currently $a ——————————————————————————————————————————————— Print out a
   ((a++)) —————————————————————————————————————————————————————————————— Increment a
done ———————————————————————————————————————————————————————————————————— End the while loop

echo ———————————————————————————————————————————————————————————————————— Print out an empty line

# Read each line in example2.fasta, find the character count of that line,
# add it to the sum, and print it all out.
while read line ————————————————————————————————————————————————————————— Begin the while loop, reading each line
do —————————————————————————————————————————————————————————————————————— Do the following commands
     chars=$(echo $line | wc -c) ———————————————————————————————————————— Find the character count of the current line
     sum1=$((sum1+chars)) ——————————————————————————————————————————————— Add the character count ot the sum and save it as the sum

     echo The sum of all the characters in the file is $sum1————————————— Print out the sum
done < example2.fasta ——————————————————————————————————————————————————— End the while loop, taking in example2.fasta as input

echo ———————————————————————————————————————————————————————————————————— Print out an empty line
```

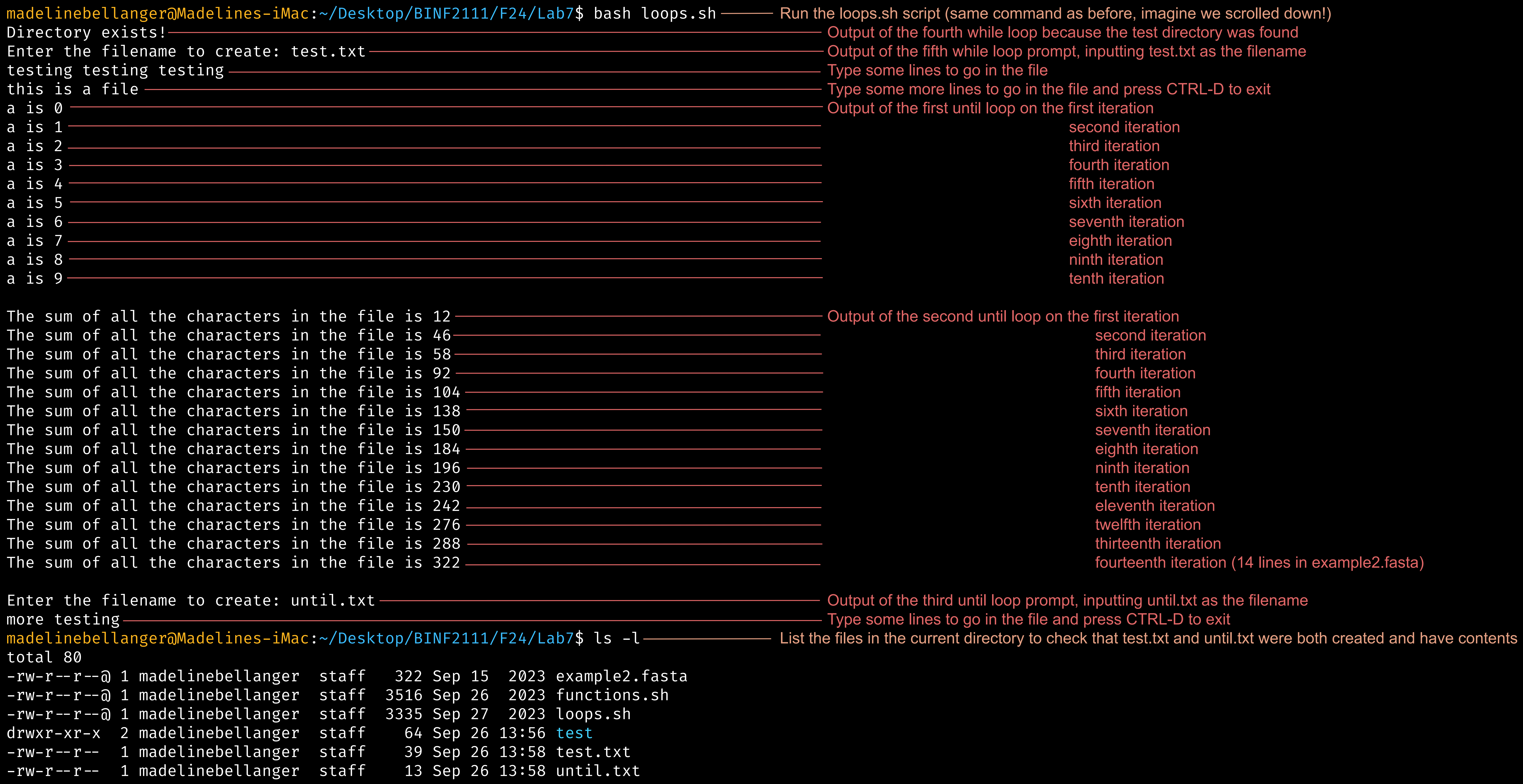


```
madelinebellanger@Madelines-iMac:~/Desktop/BINF2111/F24/Lab7$ cat loops.sh                 Print the contents of functions.sh (same command as before, imagine we scrolled down!)
#─────────────────── Cool While Loop Uses ───────────────────

# BREAKS: End the while loop when the user enters -1, otherwise keep
# adding two numbers
while :                                                                                   Begin the while loop
do                                                                                        Do the following commands
     read -p "Enter two numbers (-1 to quit): " a b                                       Read in two numbers, storing them as a and b, after prompting the user

     if [[ $a -eq -1 ]]                                                                   Check if a is equal to -1
     then                                                                                 Do the following commands
          break                                                                           Break out of the loop
     fi                                                                                   End the if statement

     ans=$((a+b))                                                                         Add a and b together and store it as ans
     echo The sum is "$ans"                                                               Print out ans
done                                                                                      End the while loop

echo                                                                                      Print out an empty line

# SLEEP: If the directory is not found, print the date and "still waiting"
# and wait for 3 seconds. If the while loop finishes, print that the
# directory was found (Ctrl-C to exit if it doesn't finish)
directory_expected="test"                                                                 Set the expected directory as test

while [[ ! -d $directory_expected ]]                                                      Begin the while loop, checking if the expected directory exists
do                                                                                        Do the following commands
   echo "`date` - Still waiting"                                                          Print out the date and "Still waiting"
   sleep 3                                                                                Wait for 3 seconds
done                                                                                      End the while loop

echo "Directory exists!"                                                                  Print out a statement that the directory exists

# READ: Write content into a file. Press Enter, Ctrl-D when you are done
# typing the file contents
echo -n "Enter the filename to create: "                                                  Print out a statement to the user
read filename            # Take the filename that will be created                        Read in the filename typed by the user

while read line     # Read the content of the file from the terminal                      Read the contents typed by the user
do                                                                                        Do the following commands
    echo $line >> $filename                                                               Print each line into the file given by the user
done                                                                                      End the while loop

# INFINITE: Press Ctrl+C to get out of the loop/end the script
#while :                                                                                  Begin the while loop
#do                                                                                       Do the following commands
#  echo "An Infinite loop"                                                                Print out a statement
#done                                                                                     End the while loop
```

```
madelinebellanger@Madelines-iMac:~/Desktop/BINF2111/F24/Lab7$ cat loops.sh ——— Print the contents of functions.sh (same command as before, imagine we scrolled down!)
#———————————— UNTIL LOOPS ————————————

# Until a is NOT less than 10, print a and add 1
a=0 ——————————————————————————————————————————————— Set a to 0
until [[ ! $a -lt 10 ]]———————————————————————————— Begin the until loop, checking if a is NOT less than 10
do ———————————————————————————————————————————————— Do the following commands
   echo a is $a ——————————————————————————————————— Print out a
   ((a++)) ———————————————————————————————————————— Increment a
done —————————————————————————————————————————————— End the until loop

echo —————————————————————————————————————————————— Print out an empty line

#———————————— Cool Until Loop Uses ————————————

# INFINITE LOOP: Until condition is true, print the iteration number,
# increment the iteration number, and wait 1 second. This loop will never
# end since the condition is hard coded to false. Press Ctrl-C to end the
# loop
con=false ————————————————————————————————————————— Set the con variable to false
itnum=0 ——————————————————————————————————————————— Set the itnum variable to 0
#until $con ——————————————————————————————————————— Begin the until loop, checking if con is true
#do ——————————————————————————————————————————————— Do the following commands
#    echo "Iteration no: $itnum"——————————————————— Print the itnum
#    ((itnum++)) —————————————————————————————————— Increment itnum
#    sleep 1 —————————————————————————————————————— Wait for 1 second
#done —————————————————————————————————————————————— End the until loop

# Read each line in example2.fasta, find the character count of that line,
# add it to the sum, and print it all out. Has the exact same
# functionality as the while loop shown above
until ! read line —————————————————————————————————— Begin the until loop, checking to see if there are still lines to be read
do ————————————————————————————————————————————————— Do the following commands
    chars2=$(echo $line | wc -c) ——————————————————— Find the character count of the current line
    sum2=$((sum2+chars2)) —————————————————————————— Add the character count ot the sum and save it as the sum

    echo The sum of all the characters in the file is $sum2 ——— Print out the sum
done < example2.fasta —————————————————————————————— End the until loop, taking in example2.fasta as input

# READ: Write content into a file. Press Enter, Ctrl-D when you are done
# typing the file contents
echo -n "Enter the filename to create: "———————————— Print out a statement to the user
read filename          # Take the filename that will be created ——— Read in the filename typed by the user

until ! read line      # Read the content of the file from the terminal ——— Read the contents typed by the user (check that there are no lines left)
do ————————————————————————————————————————————————— Do the following commands
   echo $line >> $filename ————————————————————————— Print each line into the file given by the user
done ——————————————————————————————————————————————— End the until loop
```



```
madelinebellanger@Madelines-iMac:~/Desktop/BINF2111/F24/Lab7$ mkdir test          Make a new directory called test (so we don't have an infinite loop)
madelinebellanger@Madelines-iMac:~/Desktop/BINF2111/F24/Lab7$ bash loops.sh       Run the loops.sh script
The sum of all the numbers thus far: 1           Output of the first for loop on the first iteration, 0 + 1 = 1
The sum of all the numbers thus far: 3                                  second iteration, 1 + 2 = 3
The sum of all the numbers thus far: 6                                  third iteration, 3 + 3 = 6
The sum of all the numbers thus far: 10                                 fourth iteration, 6 + 4 = 10
The sum of all the numbers thus far: 15                                 fifth iteration, 10 + 5 = 15
The sum of all the numbers thus far: 21                                 sixth iteration, 15 + 6 = 21
The sum of all the numbers thus far: 28                                 seventh iteration, 21 + 7 = 28
The sum of all the numbers thus far: 36                                 eighth iteration, 28 + 8 = 36
The sum of all the numbers thus far: 45                                 ninth iteration, 36 + 9 = 45
The sum of all the numbers thus far: 55                                 tenth iteration, 45 + 10 = 55

Item length is 7                                 Output of the second for loop on the first iteration (array item 1)
Item length is 7                                                         second iteration (array item 2)
Item length is 7                                                         third iteration (array item 3)

a is currently 0                                 Output of the first while loop on the first iteration
a is currently 1                                                        second iteration
a is currently 2                                                        third iteration
a is currently 3                                                        fourth iteration
a is currently 4                                                        fifth iteration
a is currently 5                                                        sixth iteration
a is currently 6                                                        seventh iteration
a is currently 7                                                        eighth iteration
a is currently 8                                                        ninth iteration
a is currently 9                                                        tenth iteration

The sum of all the characters in the file is 12  Output of the second while loop on the first iteration
The sum of all the characters in the file is 46                          second iteration
The sum of all the characters in the file is 58                          third iteration
The sum of all the characters in the file is 92                          fourth iteration
The sum of all the characters in the file is 104                         fifth iteration
The sum of all the characters in the file is 138                         sixth iteration
The sum of all the characters in the file is 150                         seventh iteration
The sum of all the characters in the file is 184                         eighth iteration
The sum of all the characters in the file is 196                         ninth iteration
The sum of all the characters in the file is 230                         tenth iteration
The sum of all the characters in the file is 242                         eleventh iteration
The sum of all the characters in the file is 276                         twelfth iteration
The sum of all the characters in the file is 288                         thirteenth iteration
The sum of all the characters in the file is 322                         fourteenth iteration (14 lines in example2.fasta)

Enter two numbers (-1 to quit): 2 5              Output of the prompt in the third while loop on the first iteration, inputting 2 and 5
The sum is 7                                     Sum of the inputs (2+5=7)
Enter two numbers (-1 to quit): 9 7              Output of the prompt in the third while loop on the second iteration, inputting 9 and 7
The sum is 16                                    Sum of the inputs (9+7=16)
Enter two numbers (-1 to quit): -1               Output of the prompt in the third while loop on the third iteration, inputting -1 to quit
```



```
madelinebellanger@Madelines-iMac:~/Desktop/BINF2111/F24/Lab7$ bash loops.sh
Directory exists!
Enter the filename to create: test.txt
testing testing testing
this is a file
a is 0
a is 1
a is 2
a is 3
a is 4
a is 5
a is 6
a is 7
a is 8
a is 9

The sum of all the characters in the file is 12
The sum of all the characters in the file is 46
The sum of all the characters in the file is 58
The sum of all the characters in the file is 92
The sum of all the characters in the file is 104
The sum of all the characters in the file is 138
The sum of all the characters in the file is 150
The sum of all the characters in the file is 184
The sum of all the characters in the file is 196
The sum of all the characters in the file is 230
The sum of all the characters in the file is 242
The sum of all the characters in the file is 276
The sum of all the characters in the file is 288
The sum of all the characters in the file is 322

Enter the filename to create: until.txt
more testing
madelinebellanger@Madelines-iMac:~/Desktop/BINF2111/F24/Lab7$ ls -l
total 80
-rw-r--r--@ 1 madelinebellanger  staff   322 Sep 15  2023 example2.fasta
-rw-r--r--@ 1 madelinebellanger  staff  3516 Sep 26  2023 functions.sh
-rw-r--r--@ 1 madelinebellanger  staff  3335 Sep 27  2023 loops.sh
drwxr-xr-x  2 madelinebellanger  staff    64 Sep 26 13:56 test
-rw-r--r--  1 madelinebellanger  staff    39 Sep 26 13:58 test.txt
-rw-r--r--  1 madelinebellanger  staff    13 Sep 26 13:58 until.txt
```

Run the loops.sh script (same command as before, imagine we scrolled down!)
Output of the fourth while loop because the test directory was found
Output of the fifth while loop prompt, inputting test.txt as the filename
Type some lines to go in the file
Type some more lines to go in the file and press CTRL-D to exit
Output of the first until loop on the first iteration
second iteration
third iteration
fourth iteration
fifth iteration
sixth iteration
seventh iteration
eighth iteration
ninth iteration
tenth iteration

Output of the second until loop on the first iteration
second iteration
third iteration
fourth iteration
fifth iteration
sixth iteration
seventh iteration
eighth iteration
ninth iteration
tenth iteration
eleventh iteration
twelfth iteration
thirteenth iteration
fourteenth iteration (14 lines in example2.fasta)

Output of the third until loop prompt, inputting until.txt as the filename
Type some lines to go in the file and press CTRL-D to exit
List the files in the current directory to check that test.txt and until.txt were both created and have contents