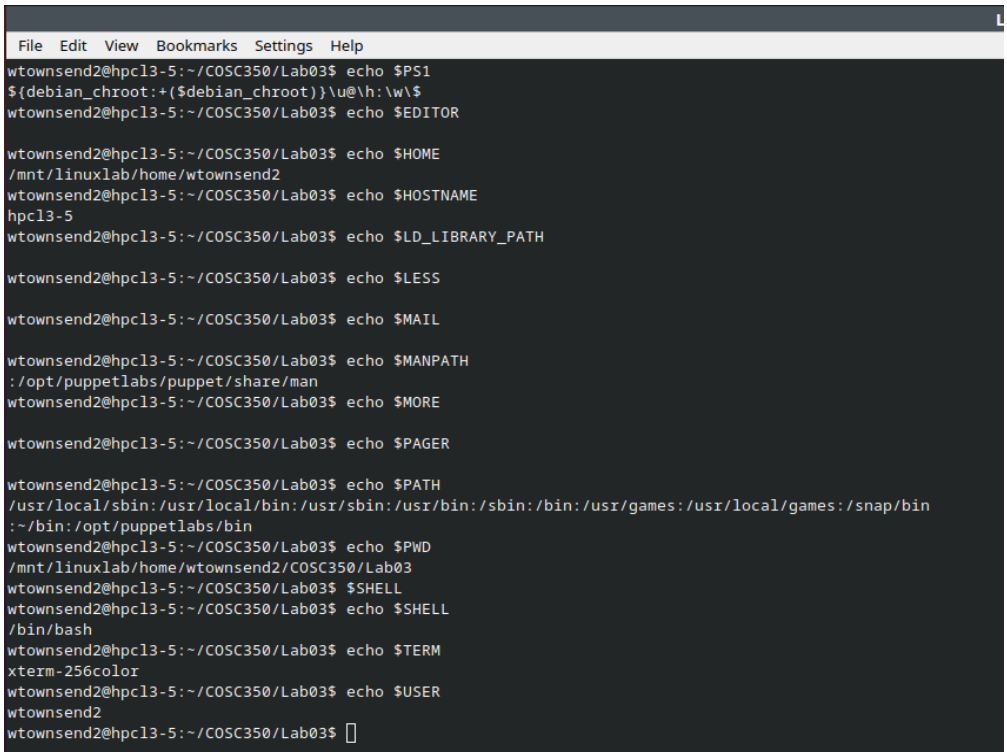
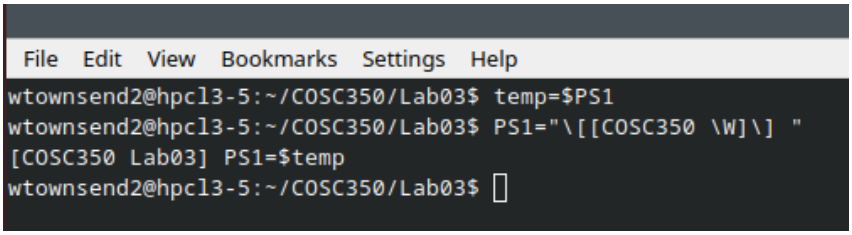
Will Townsend

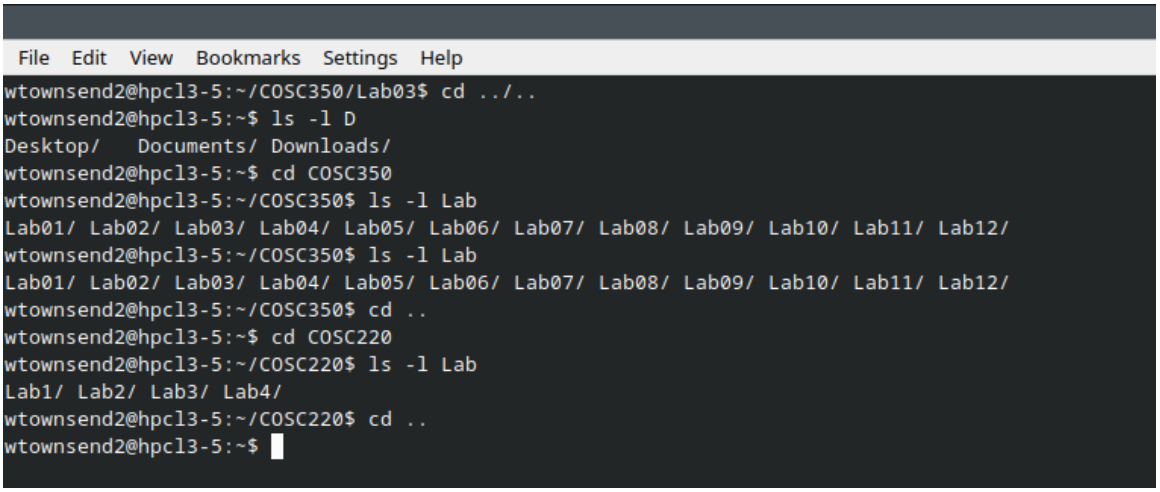
Task 1:
I tested every variable as seen below and the ones that are empty make logical sense.



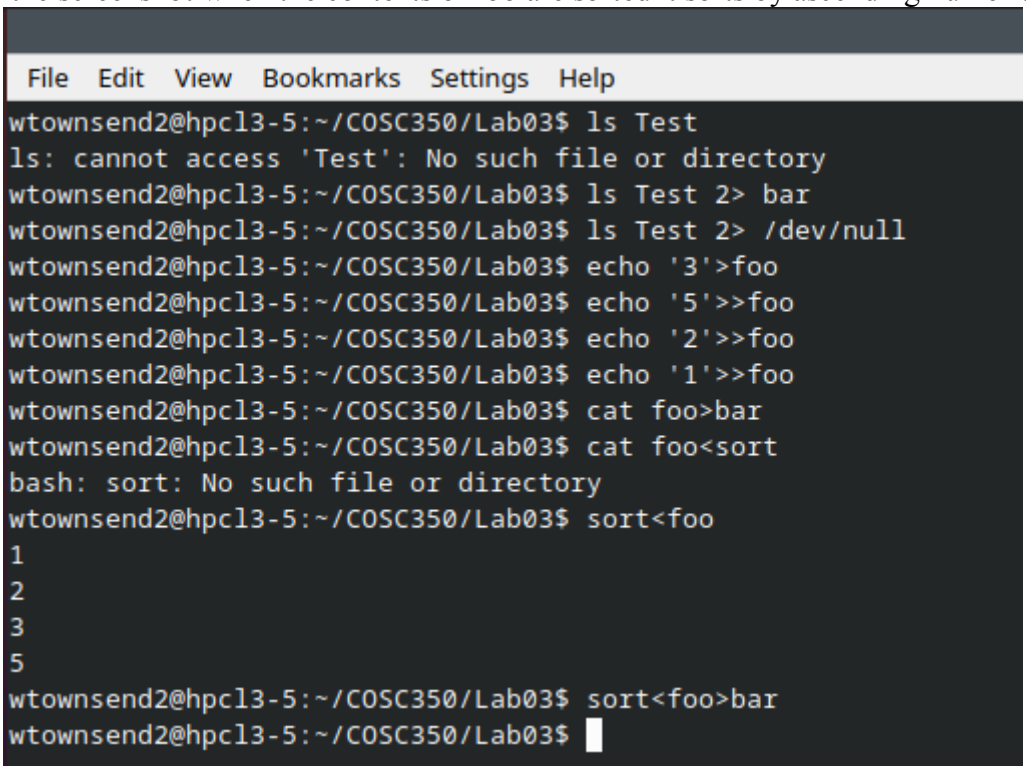
Task 2:



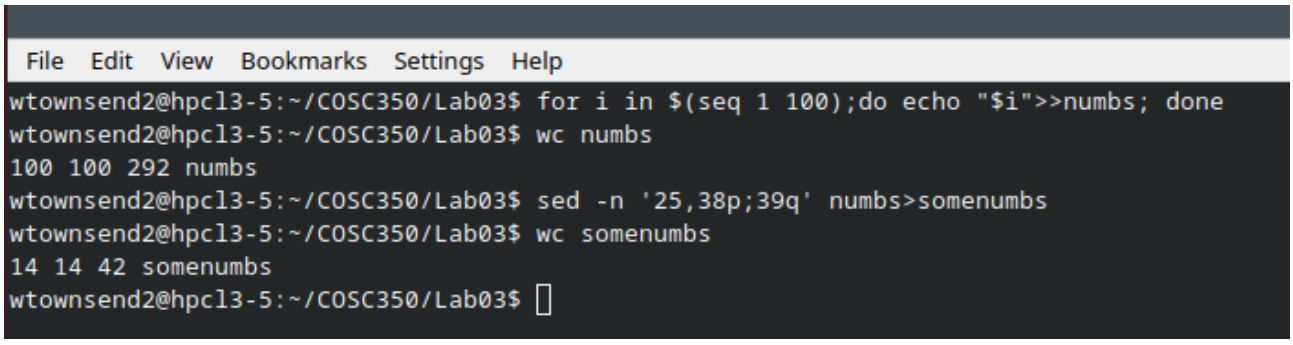
Task 3:



Task 4:
As seen in the screenshot when the contents of foo are sorted it sorts by ascending numerical order.



Task 5:
wc outputs the following information: print newline count, word count and byte count so 100 new lines 100 ‘words’ and 292 byte count in numbs and 14 newline count, 14 word count, and 42 byte count



Task 6a:

```
#!/bin/bash

base=0
num=0

while [ $base -lt 4 ] || [ $num -eq 0 ];do
   echo -n "Base Value: "
   read base
   num=`expr $base % 2`
   if [ $base -lt 4 ];then
      echo "Base Value too small!"
   elif  [ $num -eq 0 ];then
      echo "Base Value must be odd!"
   fi
done

half=`expr $base / 2`
for i in `seq $half -1 1`;do
   for j in `seq 1 $i`;do
      echo -n ' '
   done
   star=`expr $half - $i`
   star=`expr 2 \* $star + 1`
   for j in `seq 1 $star`;do
      echo -n '*'
   done
   echo ' '
done
for i in `seq 1 $base`;do
   echo -n '*'
done
echo
for i in `seq 1 $half`;do
   for j in `seq 1 $i`;do
      echo -n ' '
   done
   star=`expr $half - $i`
   star=`expr 2 \* $star + 1`
   for j in `seq 1 $star`;do
      echo -n '*'
   done
   echo ' '
done
echo
exit 0
```

Task 6b:

```
#!/bin/bash

base=0
num=0
tail=0

while [ $base -lt 4 ] || [ $num -eq 0 ];do
   echo -n "Base Value: "
   read base
   num=`expr $base % 2`
   if [ $base -lt 4 ];then
      echo "Base Value too small!"
   elif  [ $num -eq 0 ];then
      echo "Base Value must be odd!"
   fi
done

while [ $tail -le 0 ];do
   echo -n "Tail value: "
   read tail
   if [ $tail -le 0 ];then
      echo "Tail value must be positive!"
   fi
done

tail=`expr $base + $tail`
half=`expr $base / 2`
for i in `seq  1 $half`;do
space=`expr $base - 2 \* $i`
   for j in `seq 1 $space`;do
      echo -n ' '
   done
   star=`expr 2 \* $i - 1`
   for j in `seq 1 $star`;do
      echo -n '*'
   done
   echo
done
for i in `seq 1 $tail`;do
   echo -n '*'
done
echo
for i in `seq $half -1 1`;do
   space=`expr $base - 2 \* $i`
   for j in `seq 1 $space`;do
      echo -n ' '
   done
   star=`expr 2 \* $i - 1`
   for j in `seq 1 $star`;do
      echo -n '*'
   done
```

```
    echo
done
exit 0
```

Task 6c:

```
#!/bin/bash

base=0
num=0
tail=0

while [ $base -lt 4 ] || [ $num -eq 0 ];do
    echo -n "Base Value: "
    read base
    num=`expr $base % 2`
    if [ $base -lt 4 ];then
        echo "Base Value too small!"
    elif  [ $num -eq 0 ];then
        echo "Base Value must be odd!"
    fi
done

while [ $tail -le 0 ];do
    echo -n "Tail value: "
    read tail
    if [ $tail -le 0 ];then
        echo "Tail value must be positive!"
    fi
done

half=`expr $base / 2 - 1`
for i in `seq 0 $half`;do
    for j in `seq 1 $tail`;do
        echo -n ' '
    done
    star=`expr 2 \* $i + 1`
    for j in `seq 1 $star`;do
        echo -n '*'
    done
    echo
done
mid=`expr $tail + $base`
for i in `seq 1 $mid`;do
    echo -n '*'
done
echo
for i in `seq $half -1 0`;do
    for j in `seq 1 $tail`;do
        echo -n ' '
    done
    star=`expr 2 \* $i + 1`
    for j in `seq 1 $star`;do
    echo -n '*'
    done
    echo
done
```

```
exit 0
```

Task 7:

```
#!/bin/bash

echo "Task 7:"
echo -n "Enter a value: "
read num

echo -n "$num!="
res=1

while [ $num -gt 1 ]
do
    res=`expr $res \* $num`
    let num--
done
echo "$res"
exit 0
```

Task 8:

```
#!/bin/bash

echo "Task 8:"
res=0
if [ "$#" -eq 1 ];then
    len=`expr length $1`
    for i in `seq 1 $len`;do
        digit=`expr substr $1 $i 1`
        res=`expr $res + $digit`
    done
    echo "The sum of each digit in $1 is $res"
else
    echo "Must pass one parameter!"
fi
exit 0
```

Task 9:

```
#!/bin/bash

echo "Task 9:"
echo -n "Please enter the directory path (path/to/file/from/root no ~): "
read dir
path=$PWD
cd /
if [ -d $dir ];then
   cd $dir
   num=3
   while [ $num -gt 0 ];do
      echo -n "Please enter the file name: "
      read file
      if [ -f $file ] && [ -r $file ];then
         echo -n "Please enter a word to search for: "
         read word
         if grep -q $word $file
         then
            echo "$word was found!"
            cd /
            cd $path
            exit 0
         else
            echo "$word not found!"
            cd /
            cd $path
            exit 4
         fi
      else
         let num--
         echo "File does not exist or is not readable ($num more attempts)."
      fi
      if [ ! -f $file ] && [ $num -eq 0 ];then
         cd $path
         exit 2
      elif [ -r $file ] && [ $num -eq 0 ];then
         cd $path
         exit 3
      fi
   done
   exit 2
else
   echo "This is not an existing directory."
   cd $path
   exit 1
fi
```