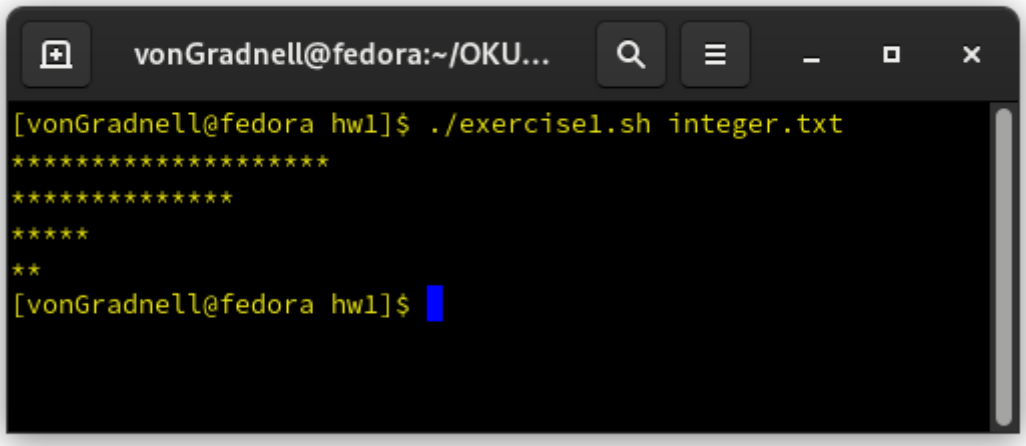
CENG322 ASSIGNMENT1
REPORT

Exercise 1:

Implementation:

In the code, firstly I passed the argument to FILENAME variable. Then I read the file line by line and in the for loop, I prompted * as many as the number in the line without spaces. When for loop ended, I called 'echo' command to switch the new line. As we can see in the line 14, I passed the FILENAME variable to while loop.

Output:



Exercise 2:

Implementation:

```
echo "Enter a sequence of numbers followed by 'end'"
MAX=0
VALID=true
i=1
RE='^[0-9]+$'
```

Firstly I prompted instruction and I created some useful variables.

```
# infinite loop until break
while [ $VALID ]
do
  # take input
  read NUM
```

Then I wrote a while loop for taking inputs until the user exits.

```
# check if input is 'end'
# if it is, break loop
if [ $NUM = 'end' ];
then
  break
```

In while loop, I checked if the input is equal to 'end', if it is then I broke the loop.

```
# check if input is integer
# if it is not, break loop
elif ! [[ $NUM =~ $RE ]];
then
  echo "Please enter only integer or 'end' !!"
  break
fi
```

Then I checked if the input is not an integer. If it is not an integer then I warned the user and broke the loop.

```
# if the first input is entered
# then it is max number
if [ $i -eq 1 ]
then
  MAX=$NUM
```

After the checking input, I checked if input is the first one. If input is the user's first input then I assigned to MAX variable.

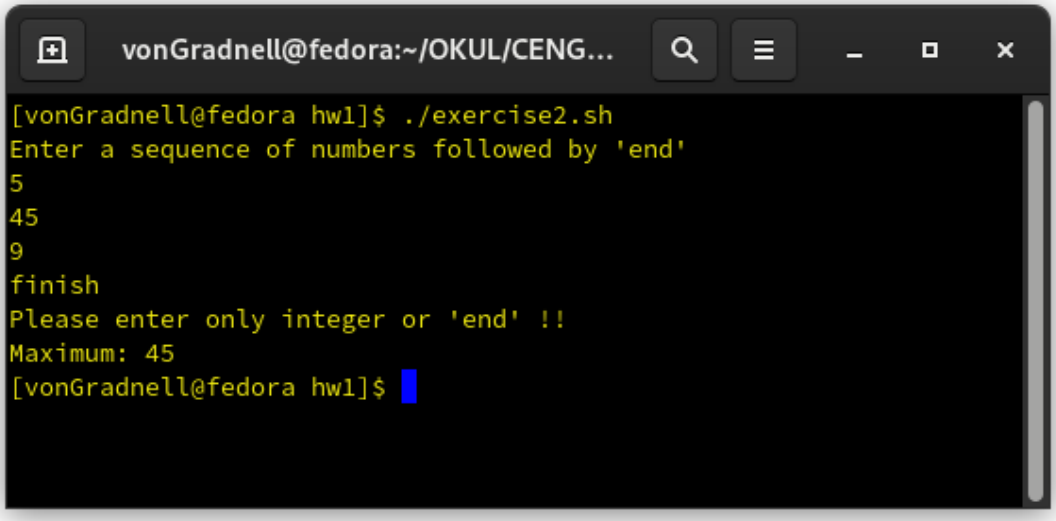
```
else
  # check if input is greater
  # than max number
  # if it is, assign to max
  if [ $NUM -gt $MAX ]
  then
    MAX=$NUM
  fi
fi
```

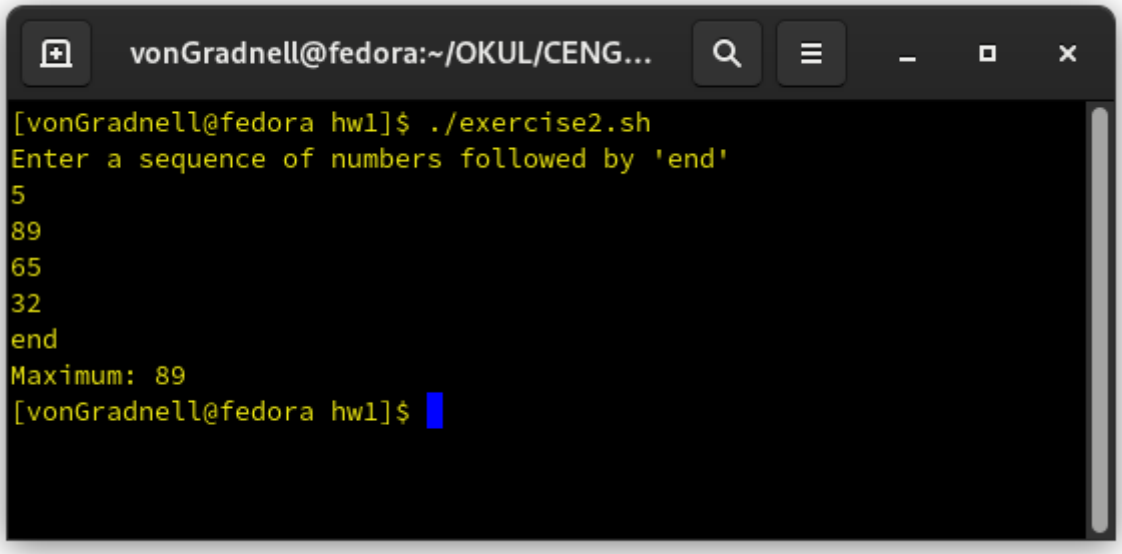
If input is not first input, then I compared the input with MAX variable. If it is greater than MAX variable, then I assigned input number to MAX.

```
  # increment input number
  i=$((i+1))
done

echo "Maximum: $MAX"
```

In while loop I also increment the input number to memorize how many input entered the user. After the while loop, I prompted the MAX variable.

Outputs:





Exercise 3:

Implementation:

```
DIR=""
#check if there is an argument
if [ $# -eq 0 ]
#if there is not, then working
#directory is assigned to variable
then
  DIR=$(pwd)
#if there is an argument
```

Firstly I checked if there is an argument from the user. If there is not, our directory, DIR variable, is current directory.

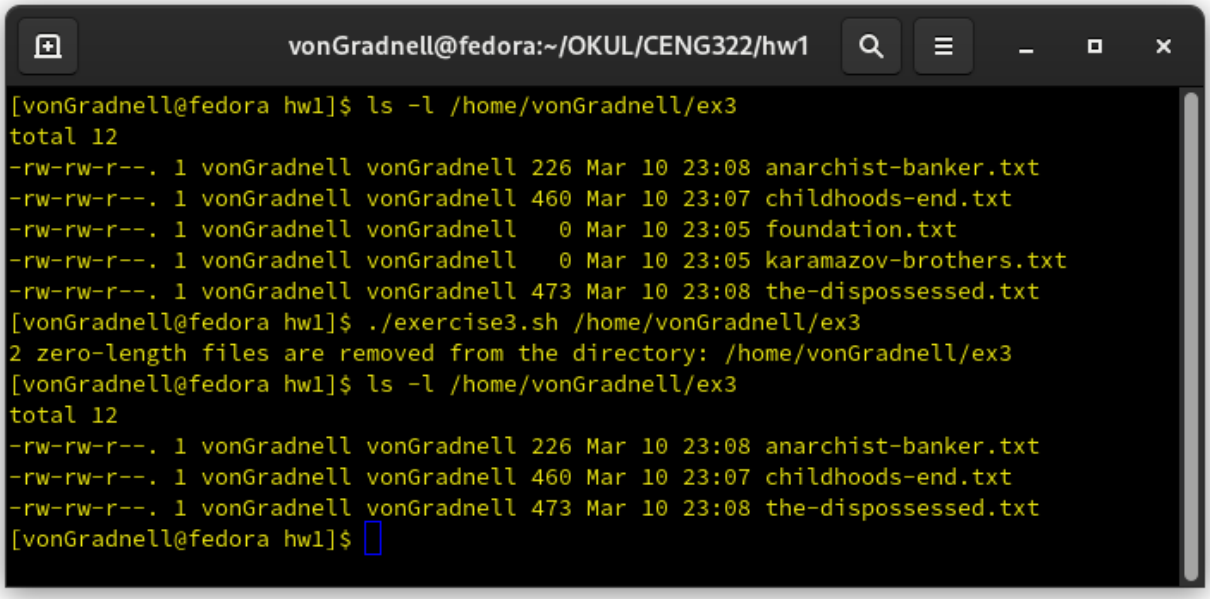
```
#if there is an argument
else
  #check if argument is valid
  if [ ! -d $1 ]
  #if not valid
  then
    echo "Error: '$1' not found."
    exit 1
  else
  #if valid, assign to variable
    DIR=$1
  fi
fi
```

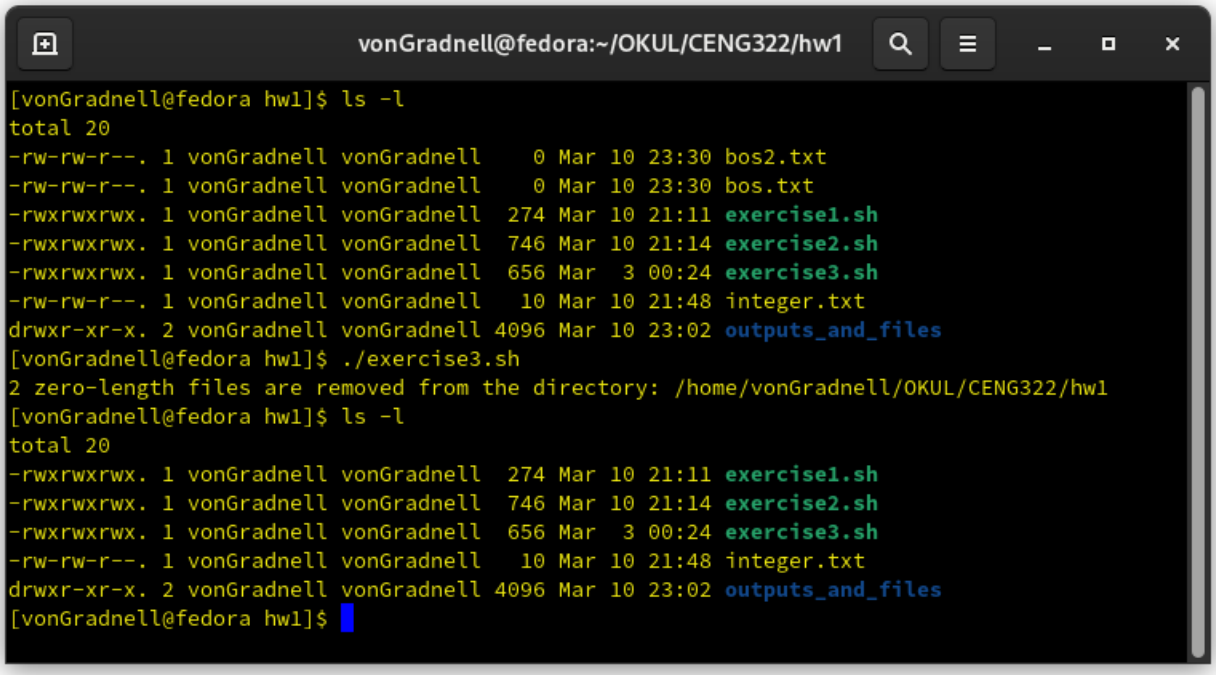
If there is and argument, then I checked if it is valid, if there is a directory with the given name in the argument. If it is not valid then exit with error. If it is valid then assigned to variable DIR.

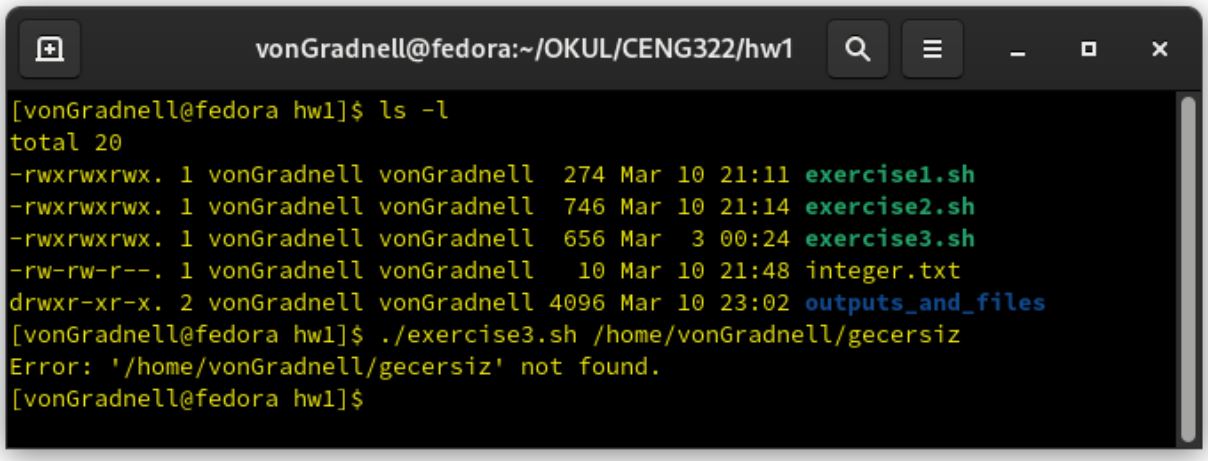
```
REMOVED=0

for FILE in $( ls $DIR ) #iterate all files in directory
do
  if [ -f $DIR/$FILE ] && [ ! -s $DIR/$FILE ] #if file exists and is empty
  then
    rm -rf $DIR/$FILE #remove the file
    REMOVED=$((REMOVED+1))
  fi
done

echo "$REMOVED zero-length files are removed from the directory: $DIR"
```

In for loop I iterated all files in given directory. If file is exist and is empty I removed it and incremented REMOVED variable. Then when for loop is ended then I prompted how many files are removed from the directory.

Outputs:







Onur Cihangir
250201049