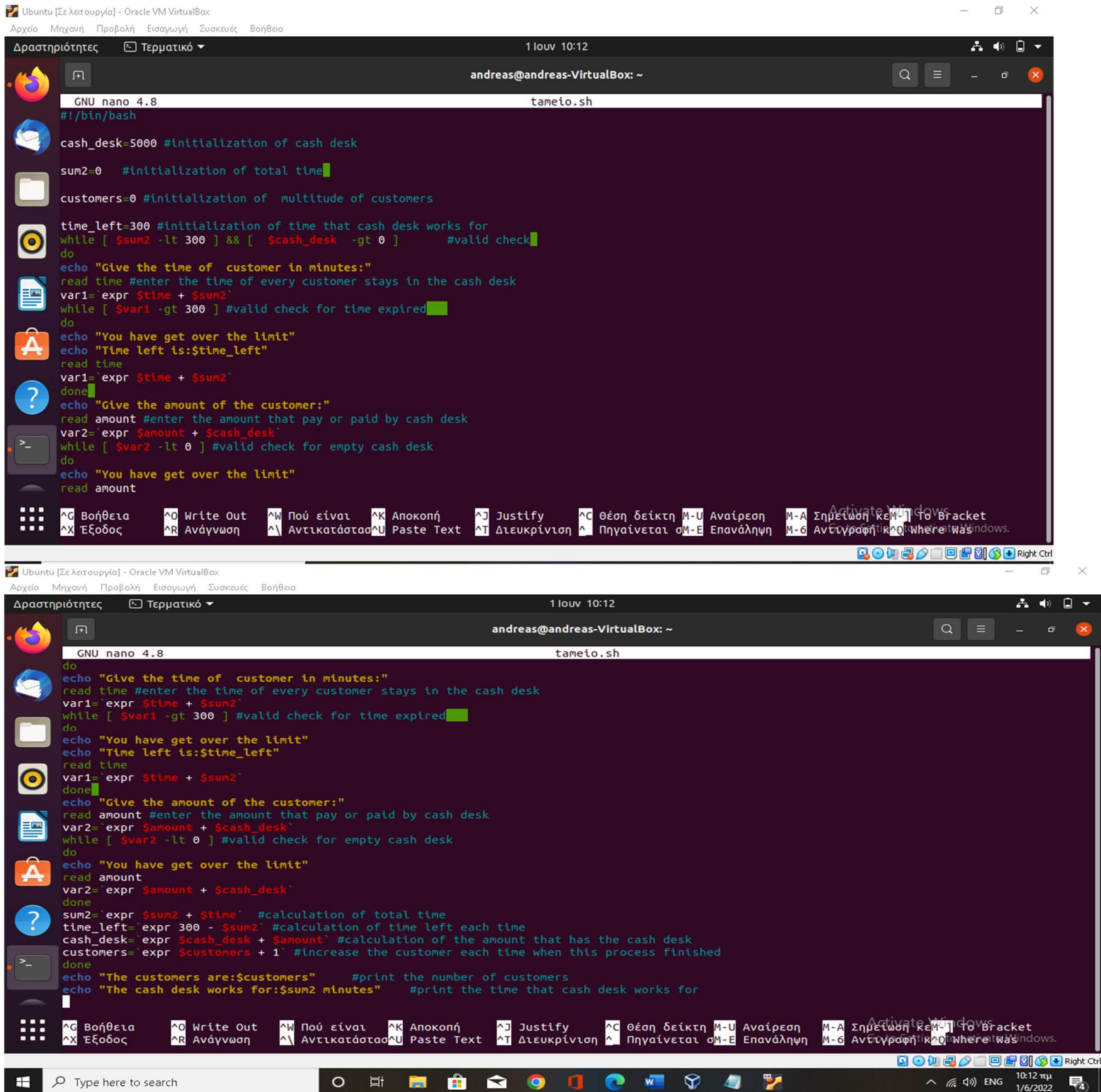


**ΕΡΓΑΣΙΑ ΕΞΑΜΗΝΟΥ**  
**ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ UNIX**  
**ΑΝΔΡΕΑΣ ΒΑΝΙΚΙΩΤΗΣ**  
**ΑΜ:Ε20013**  
**ΕΞΑΜΗΝΟ 4<sup>ο</sup>**

## Επεξηγηματικά σχόλια:

### Άσκηση 1

1. Αρχικά εμφανίζουμε τον κώδικα μας γραμμένο σε script.



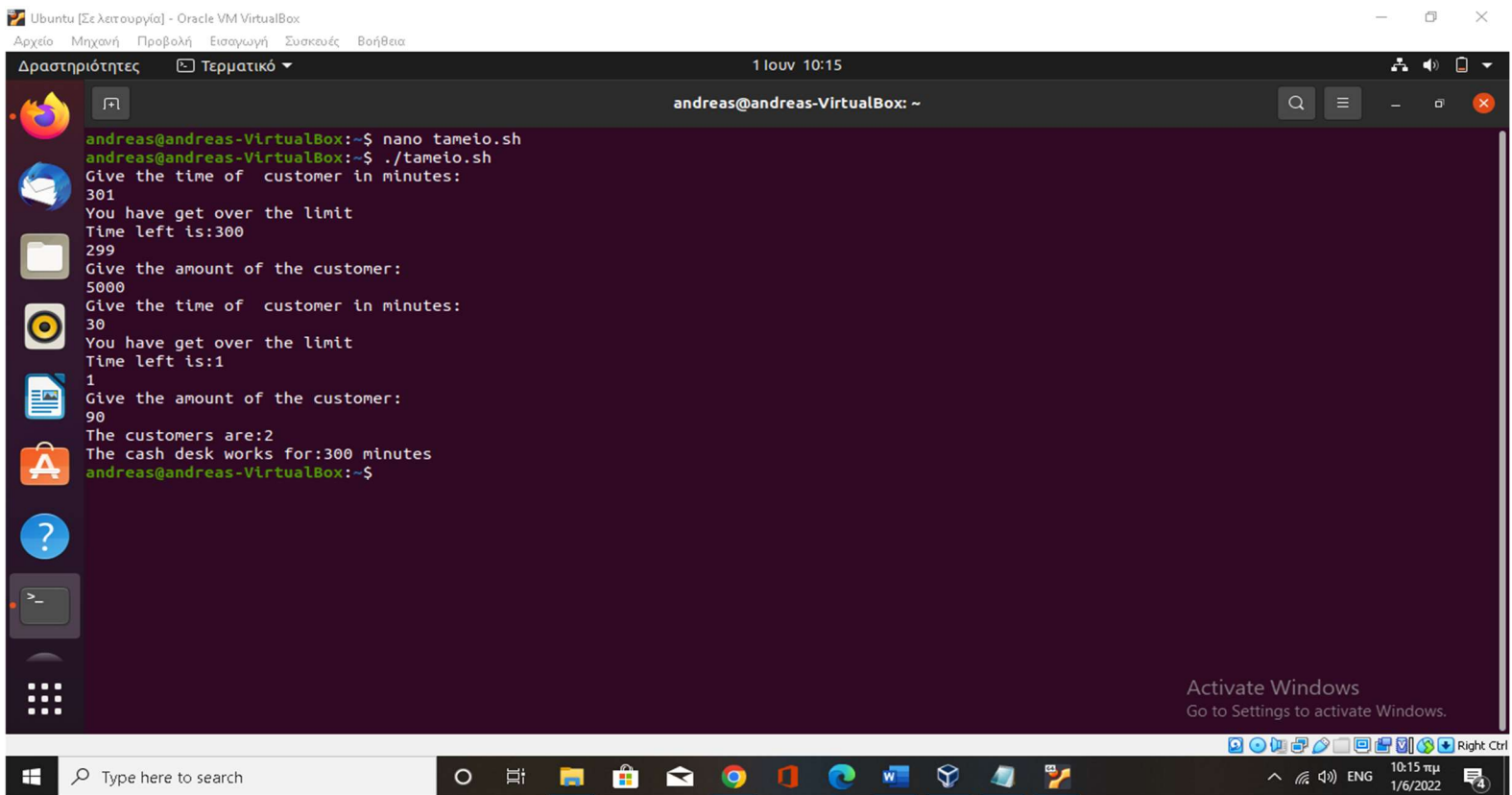
```
GNU nano 4.8 tameio.sh
#!/bin/bash

cash_desk=5000 #initialization of cash desk
sum2=0 #initialization of total time
customers=0 #initialization of multitude of customers

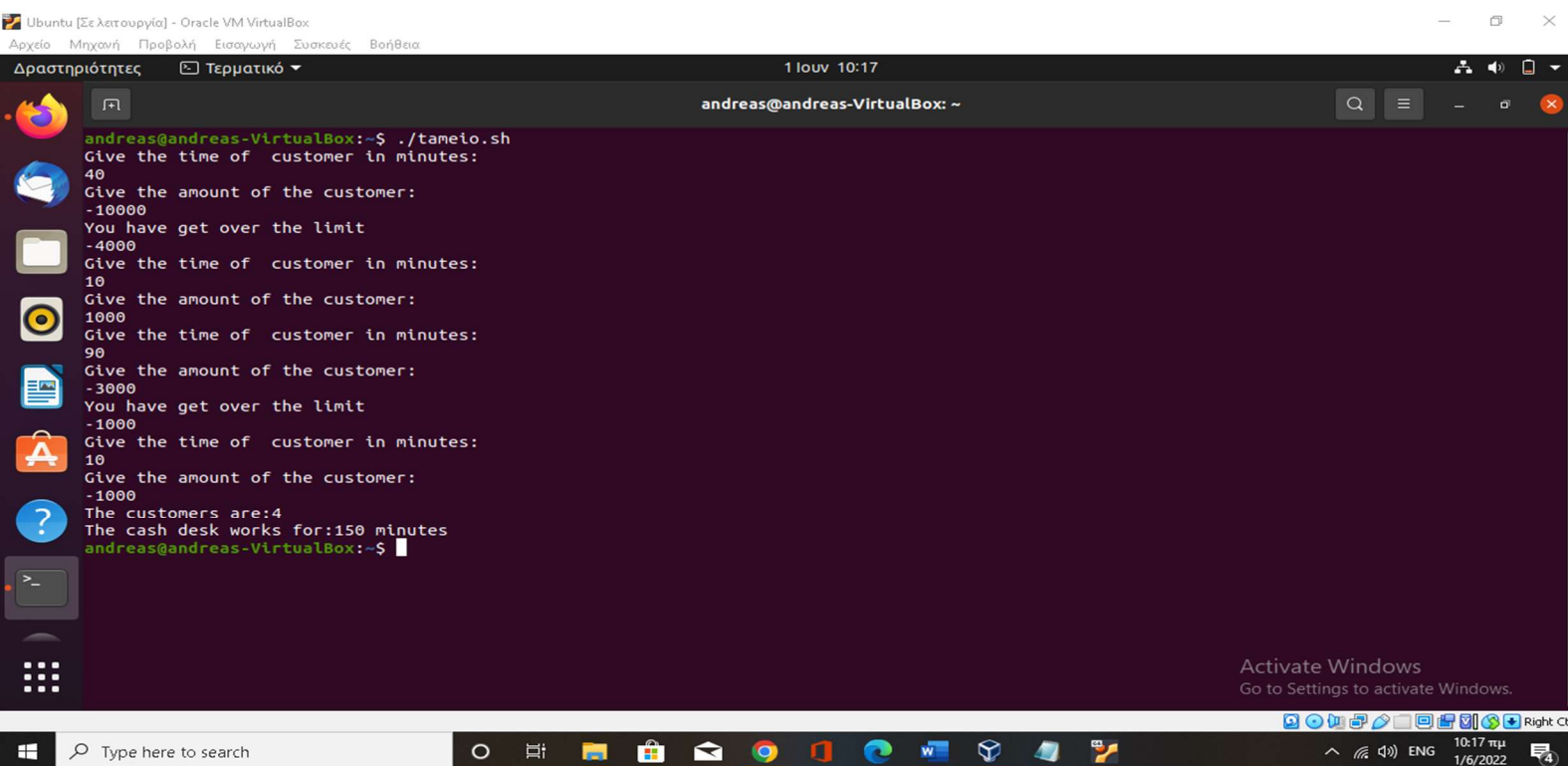
time_left=300 #initialization of time that cash desk works for
while [ $sum2 -lt 300 ] && [ $cash_desk -gt 0 ] #valid check
do
echo "Give the time of customer in minutes:"
read time #enter the time of every customer stays in the cash desk
var1=`expr $time + $sum2`
while [ $var1 -gt 300 ] #valid check for time expired
do
echo "You have get over the limit"
echo "Time left is:$time_left"
read time
var1=`expr $time + $sum2`
done
echo "Give the amount of the customer:"
read amount #enter the amount that pay or paid by cash desk
var2=`expr $amount + $cash_desk`
while [ $var2 -lt 0 ] #valid check for empty cash desk
do
echo "You have get over the limit"
read amount
done
echo "Give the time of customer in minutes:"
read time #enter the time of every customer stays in the cash desk
var1=`expr $time + $sum2`
while [ $var1 -gt 300 ] #valid check for time expired
do
echo "You have get over the limit"
echo "Time left is:$time_left"
read time
var1=`expr $time + $sum2`
done
echo "Give the amount of the customer:"
read amount #enter the amount that pay or paid by cash desk
var2=`expr $amount + $cash_desk`
while [ $var2 -lt 0 ] #valid check for empty cash desk
do
echo "You have get over the limit"
read amount
var2=`expr $amount + $cash_desk`
done
sum2=`expr $sum2 + $time` #calculation of total time
time_left=`expr 300 - $sum2` #calculation of time left each time
cash_desk=`expr $cash_desk + $amount` #calculation of the amount that has the cash desk
customers=`expr $customers + 1` #increase the customer each time when this process finished
done
echo "The customers are:$customers" #print the number of customers
echo "The cash desk works for:$sum2 minutes" #print the time that cash desk works for
```

2. Πρώτα στον κώδικα αρχικοποιούμε τις εξής τιμές: τον αριθμό χρημάτων που έχει το ταμείο, ένα άθροισμα το οποίο θα μετράει τον συνολικό χρόνο, ένα πλήθος για τον αριθμό πελατών και τέλος μια μεταβλητή η οποία θα μας ενημερώνει για τον χρόνο που απομένει.
3. Ο βασικός έλεγχος που πρέπει να κάνουμε στην επανάληψη είναι το άθροισμα του συνολικού χρόνου να μην υπερβαίνει τα 300 λεπτά τα οποία αντιστοιχούν στις 5 ώρες και το ταμείο να είναι μεγαλύτερο του 0 για να συνεχίσει να εκτελεί την επανάληψη.
4. Έπειτα διαβάζουμε τον χρόνο του κάθε πελάτη και σε μια μεταβλητή var1 θα προσθέτουμε τον συνολικό χρόνο μαζί με τον χρόνο του κάθε πελάτη και στην επαναληπτική διαδικασία θα γίνεται ένας έλεγχος εγκυρότητας σε περίπτωση που εκχωρηθεί κάποια μεγαλύτερη τιμή του 300.
5. Στην συνέχεια διαβάζουμε το ποσό που ο κάθε πελάτης πληρώνει η πληρώνεται από το ταμείο και σε μια μεταβλητή var2 ελέγχουμε αν αυτό το ποσό είναι κάτω του 0 που σημαίνει ότι το ταμείο δεν μπορεί να πληρώσει τον πελάτη και ζητάει να του ξαναεισάγει ένα άλλο ποσό.
6. Επίσης αυξάνουμε τον συνολικό χρόνο κάθε φορά, μειώνουμε τον χρόνο που απομένει, αυξομειώνουμε ανάλογα το ποσό την μεταβλητή που κρατάει το ταμείο και αυξάνουμε κάθε φορά το πλήθος των πελατών.
7. Τέλος εμφανίζουμε τον συνολικό χρόνο που λειτούργησε το ταμείο και το πλήθος των πελατών.

## Ενδεικτικά αποτελέσματα:



```
andreas@andreas-VirtualBox:~$ nano tameio.sh
andreas@andreas-VirtualBox:~$ ./tameio.sh
Give the time of customer in minutes:
301
You have get over the limit
Time left is:300
299
Give the amount of the customer:
5000
Give the time of customer in minutes:
30
You have get over the limit
Time left is:1
1
Give the amount of the customer:
90
The customers are:2
The cash desk works for:300 minutes
andreas@andreas-VirtualBox:~$
```



```
andreas@andreas-VirtualBox:~$ ./tameio.sh
Give the time of customer in minutes:
40
Give the amount of the customer:
-10000
You have get over the limit
-4000
Give the time of customer in minutes:
10
Give the amount of the customer:
1000
Give the time of customer in minutes:
90
Give the amount of the customer:
-3000
You have get over the limit
-1000
Give the time of customer in minutes:
10
Give the amount of the customer:
-1000
The customers are:4
The cash desk works for:150 minutes
andreas@andreas-VirtualBox:~$
```

## Άσκηση 2

### 1. Εμφάνιση του κώδικα μας με print screen:

The image shows three sequential screenshots of a terminal window running Ubuntu in an Oracle VM VirtualBox. The terminal displays the development of a shell script named `guess.sh` using the `nano` editor. The script is designed to guess a word based on letter frequency analysis.

**First Screenshot:** Shows the initial setup of the script, including resetting query counters, checking for arguments, and validating that arguments are positive numbers.

```
GNU nano 4.8 guess.sh
#!/bin/bash

# resets query counters
metritis1=0
metritis2=0
metritis3=0
metritis4=0
metritis5=0

# check for arguments
if [ ! $# -eq 2 ]; then
    echo "Two arguments required"
    exit 0
fi

# check for arguments to be positive numbers
if ! [[ "$1" =~ ^[0-9]+$ ]] || [ $1 -lt 0 ]
then
    echo "The first argument must be positive number"
    exit 0;
fi

if ! [[ "$2" =~ ^[0-9]+$ ]] || [ $2 -lt 0 ]
then
    echo "The second argument must be positive number"
    exit 0;
fi
```

**Second Screenshot:** Shows the script defining variables for the number of days to detect up to 2 days, and declaring arrays for pinakasI through pinakasV. It also prompts the user to enter the directory name (or E to exit) and reads the folder name.

```
GNU nano 4.8 guess.sh
# passage of definitions
dikalwmata=$1

tmeres=$((-1*$2)) # so that it can detect up to 2 days

# construction of boards with paths and meters
declare -A pinakasI
declare -A pinakasII
declare -A pinakasIII
declare -A pinakasIV
declare -A pinakasV

# expects the user to enter the name of the folder
echo -n "Fill the directory name (or E to exit) : "
read fakelos

while [ $fakelos != "E" ]; do

    # check for folder
    if [[ -d $fakelos ]]; then

        # counters of results for each sub-query
        metritisPinakaI=$(pinakasI[$fakelos])
        metritisPinakaII=$(pinakasII[$fakelos])
```

**Third Screenshot:** Shows the script defining sub-queries for counting the number of occurrences of each letter in the directory. It uses `find` and `wc` to count the occurrences of each letter and updates the counters.

```
GNU nano 4.8 guess.sh
        metritisPinakaII=$(pinakasII[$fakelos])
        metritisPinakaIII=$(pinakasIII[$fakelos])
        metritisPinakaIV=$(pinakasIV[$fakelos])
        metritisPinakaV=$(pinakasV[$fakelos])

# sub-query 1
count=$(find $fakelos -type f -perm $dikalwmata | wc -l)
echo "Ypoerwtima 1 apotelesmata= $count"

metritis1=$(( $metritis1 + $count ))
metritisPinakaI=$(( $metritisPinakaI + $count ))

find $fakelos -type f -perm $dikalwmata

echo "*****"

# sub-query 2
count=$(find $fakelos -type f -ctime $tmeres | wc -l)
echo "Ypoerwtima 2 apotelesmata= $count"

metritis2=$(( $metritis2 + $count ))
metritisPinakaII=$(( $metritisPinakaII + $count ))

find $fakelos -type f -ctime $tmeres
```



```
Ubuntu [Σε λειτουργία] - Oracle VM VirtualBox
Αρχείο Μηνυτή Προβολή Εισαγωγή Συντομίες Βοήθεια
Δραστηριότητες Τερματικό 15 Ιουν 16:48
andreas@andreas-VirtualBox: ~
GNU nano 4.8 guess.sh

find $fakelos -type f -ctime $imeres

echo "*****"

# sub-query 3
count=$(find $fakelos -type d -atime $imeres | wc -l)
echo "Ypoerwtina 3 apotelesmata= $count"

metrittis3=$(( $metrittis3 + $count ))
metrittisPinakaIII=$(( $metrittisPinakaIII + $count ))

find $fakelos -type d -atime $imeres

echo "*****"

# sub-query 4
count=$(find $fakelos -type s,p | wc -l)
echo "Ypoerwtina 4 apotelesmata= $count"

metrittis4=$(( $metrittis4 + $count ))
metrittisPinakaIV=$(( $metrittisPinakaIV + $count ))

find $fakelos -type s,p

echo "*****"

# sub-query 5
```

```
Ubuntu [Σε λειτουργία] - Oracle VM VirtualBox
Αρχείο Μηνυτή Προβολή Εισαγωγή Συντομίες Βοήθεια
Δραστηριότητες Τερματικό 15 Ιουν 16:48
andreas@andreas-VirtualBox: ~
GNU nano 4.8 guess.sh

find $fakelos -type f -ctime $imeres

echo "*****"

# sub-query 3
count=$(find $fakelos -type d -atime $imeres | wc -l)
echo "Ypoerwtina 3 apotelesmata= $count"

metrittis3=$(( $metrittis3 + $count ))
metrittisPinakaIII=$(( $metrittisPinakaIII + $count ))

find $fakelos -type d -atime $imeres

echo "*****"

# sub-query 4
count=$(find $fakelos -type s,p | wc -l)
echo "Ypoerwtina 4 apotelesmata= $count"

metrittis4=$(( $metrittis4 + $count ))
metrittisPinakaIV=$(( $metrittisPinakaIV + $count ))

find $fakelos -type s,p

echo "*****"

# sub-query 5
```

```
Ubuntu [Σε λειτουργία] - Oracle VM VirtualBox
Αρχείο Μηνυτή Προβολή Εισαγωγή Συντομίες Βοήθεια
Δραστηριότητες Τερματικό 15 Ιουν 16:53
andreas@andreas-VirtualBox: ~
GNU nano 4.8 guess.sh

# sub-query 5
count=$(find $fakelos -maxdepth 1 -type f -size 0 | wc -l)
echo "Ypoerwtina 5 apotelesmata= $count"

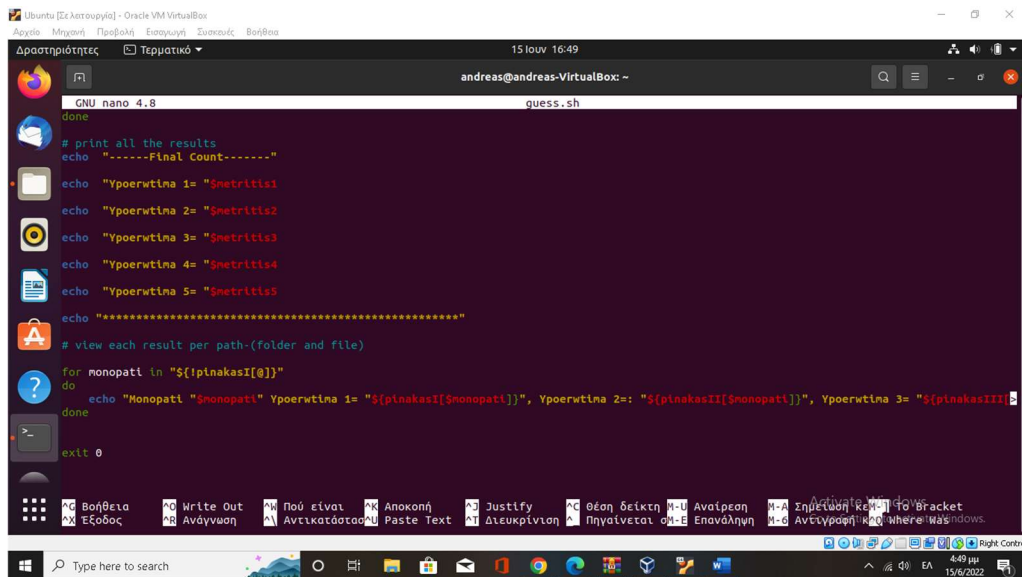
metrittis5=$(( $metrittis5 + $count ))
metrittisPinakaV=$(( $metrittisPinakaV + $count ))
find $fakelos -maxdepth 1 -type f -size 0

# transfer of results to tables
pinakasI+=$(( $fakelos ))=$metrittisPinakaI
pinakasII+=$(( $fakelos ))=$metrittisPinakaII
pinakasIII+=$(( $fakelos ))=$metrittisPinakaIII
pinakasIV+=$(( $fakelos ))=$metrittisPinakaIV
pinakasV+=$(( $fakelos ))=$metrittisPinakaV
else
echo $fakelos" cannot be processed or is not folder"
fi

# expects the user to enter the name of the folder
echo -n "Fill the directory name (or E to exit): "
read fakelos

done

# print all the results
echo "-----Final Count-----"
```



```
GNU nano 4.8 guess.sh
done
# print all the results
echo "-----Final Count-----"
echo "Ypoerwtina 1= $metritis1"
echo "Ypoerwtina 2= $metritis2"
echo "Ypoerwtina 3= $metritis3"
echo "Ypoerwtina 4= $metritis4"
echo "Ypoerwtina 5= $metritis5"
echo "*****"
# view each result per path-(folder and file)
for monopati in "${!pinakasi[@]}"
do
echo "Monopati \"$monopati\" Ypoerwtina 1= \"$pinakasi1[$monopati]\", Ypoerwtina 2= \"$pinakasi2[$monopati]\", Ypoerwtina 3= \"$pinakasi3[$monopati]\", Ypoerwtina 4= \"$pinakasi4[$monopati]\", Ypoerwtina 5= \"$pinakasi5[$monopati]\""
done
exit 0
```

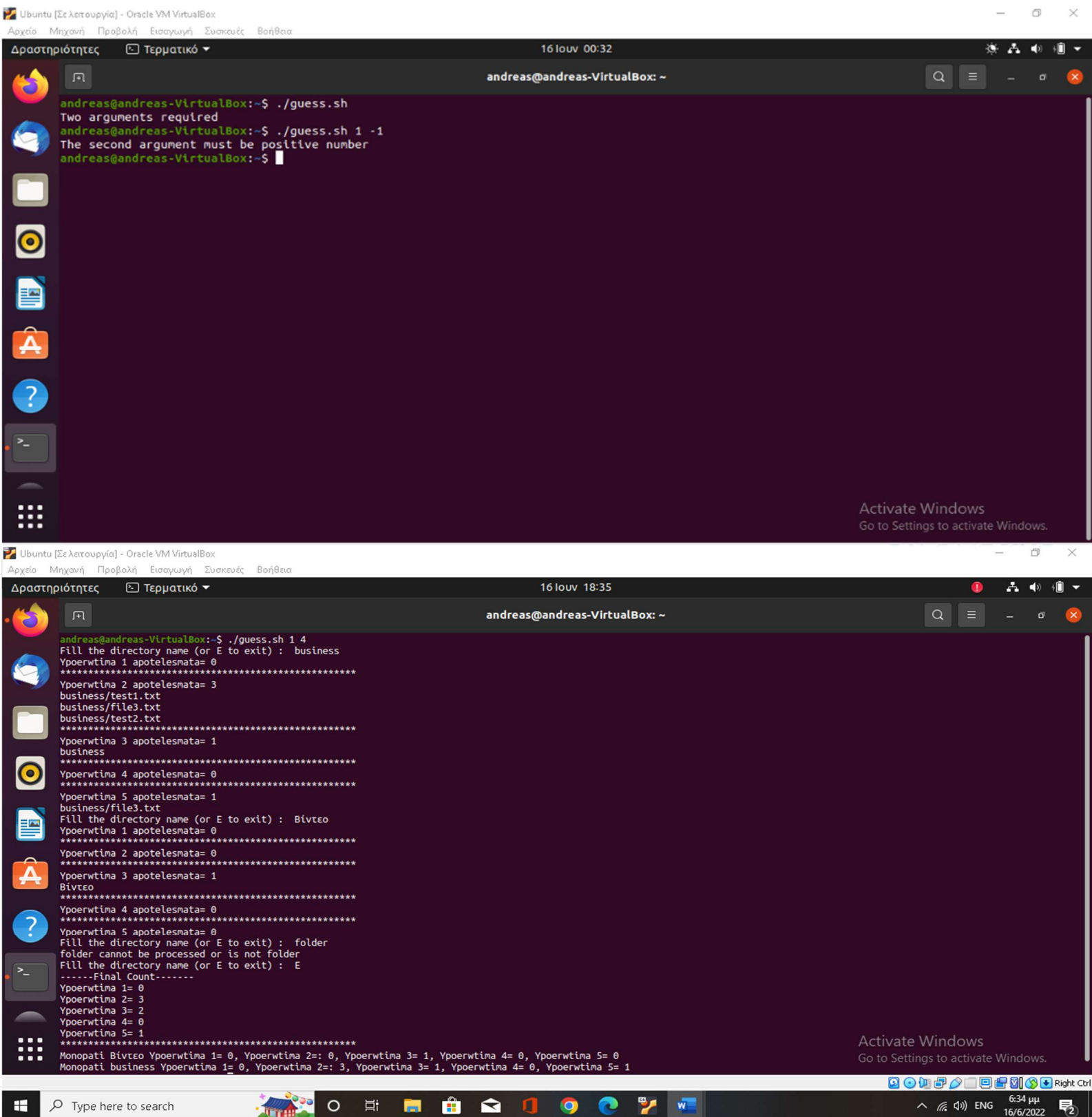
2. Αρχικά, αρχικοποιούμε τους μετρητές που θα χρησιμοποιήσουμε στα ερωτήματα.
3. Στην συνέχεια με μια δομή ελέγχου ελέγχουμε για τον αν έχουν τοποθετηθεί και τα δυο ορίσματα.
4. Αφού έχουν τοποθετηθεί ελέγχουμε ξανά αν είναι θετικοί αριθμοί αλλιώς μας εμφανίζει το κατάλληλο μήνυμα.
5. Ακολούθως γίνεται το πέρασμα των ορισμάτων(στην 1<sup>η</sup> μεταβλητή μπαίνει το 1<sup>ο</sup> όρισμα μας και στην 2<sup>η</sup> το δεύτερο όρισμα αφού έχουμε ξεπεράσει φυσικά τους προηγούμενους ελέγχους).
6. Αφότου ολοκληρωθεί και αυτό το βήμα προχωράμε στην κατασκευή των πινάκων με μονοπάτια και μετρητές και στην συνέχεια ζητάμε από τον χρήστη να βάλει το όνομα του καταλόγου που επιθυμεί.
7. Στο επόμενο βήμα μπαίνουμε σε μια επαναληπτική διαδικασία η οποία σταματάει όταν το θελήσει ο χρήστης πληκτρολογώντας το γράμμα E όποτε του ζητηθεί. Επίσης γίνεται έλεγχος για το αν ο φάκελος είναι όντως φάκελος αλλιώς εμφανίζει κατάλληλο μήνυμα. Αν είναι φάκελος κάνει τα ακόλουθα:
  - Υλοποιεί τους μετρητές των αποτελεσμάτων για κάθε υποερώτημα.
  - Υλοποίηση 1<sup>ου</sup> υποερωτήματος: Με χρήση της εντολής find βρίσκει τα αρχεία του δέντρου του δοθέντος καταλόγου με εξουσιοδοτήσεις τον αριθμό όρισμα.

- Υλοποίηση 2<sup>ου</sup> υποερωτήματος: Ξανά με χρήση της find υλοποιούμε το 2<sup>ο</sup> υποερώτημα στο οποίο βρίσκουμε τα αρχεία του δέντρου εκείνα τα οποία άλλαξαν περιεχόμενα κατά τις μέρες στις οποίες αντιστοιχεί το δεύτερο μας όρισμα.
  - Υλοποίηση 3<sup>ου</sup> υποερωτήματος: Με χρήση της find βρίσκουμε τους υποκαταλόγους που προσπελάστηκαν κατά τις ημέρες που δηλώνει το δεύτερο μας όρισμα.
  - Υλοποίηση 4<sup>ου</sup> υποερωτήματος: Με χρήση της find αυτή την φορά βρίσκουμε τα αρχεία που είναι ή τύπου socket ή τύπου pipe.
  - Υλοποίηση 5<sup>ου</sup> υποερωτήματος: Τέλος, ξαναχρησιμοποιούμε την find προκειμένου να βρούμε τα αρχεία του καταλόγου τα οποία είναι κενά
  - Μετά τις υλοποιήσεις των υποερωτημάτων μεταφέρουμε τα αποτελέσματα σε πίνακες.
  - Τέλος διαβάζουμε ξανά το όνομα του καταλόγου μέχρι ο χρήστης να επιλέξει την έξοδο.
8. Τυπώνουμε όλα τα αποτελέσματα για κάθε μετρητή.
  9. Τέλος εμφανίζουμε το κάθε αποτέλεσμα ανά path (φάκελο και αρχείο).



## Ενδεικτικά Αποτελέσματα:

Αρχικά εμφανίζουμε το script με τα λάθη. Στην πρώτη εκτέλεση βλέπουμε ότι δεν έχουμε ορίσματα και μας εμφανίζει το κατάλληλο μήνυμα όπως και στην δεύτερη εκτέλεση μας εμφανίζει το παρακάτω μήνυμα διότι ο δεύτερος αριθμός είναι αρνητικός. Στην δεύτερη εικόνα εμφανίζουμε με σωστά δεδομένα το script.

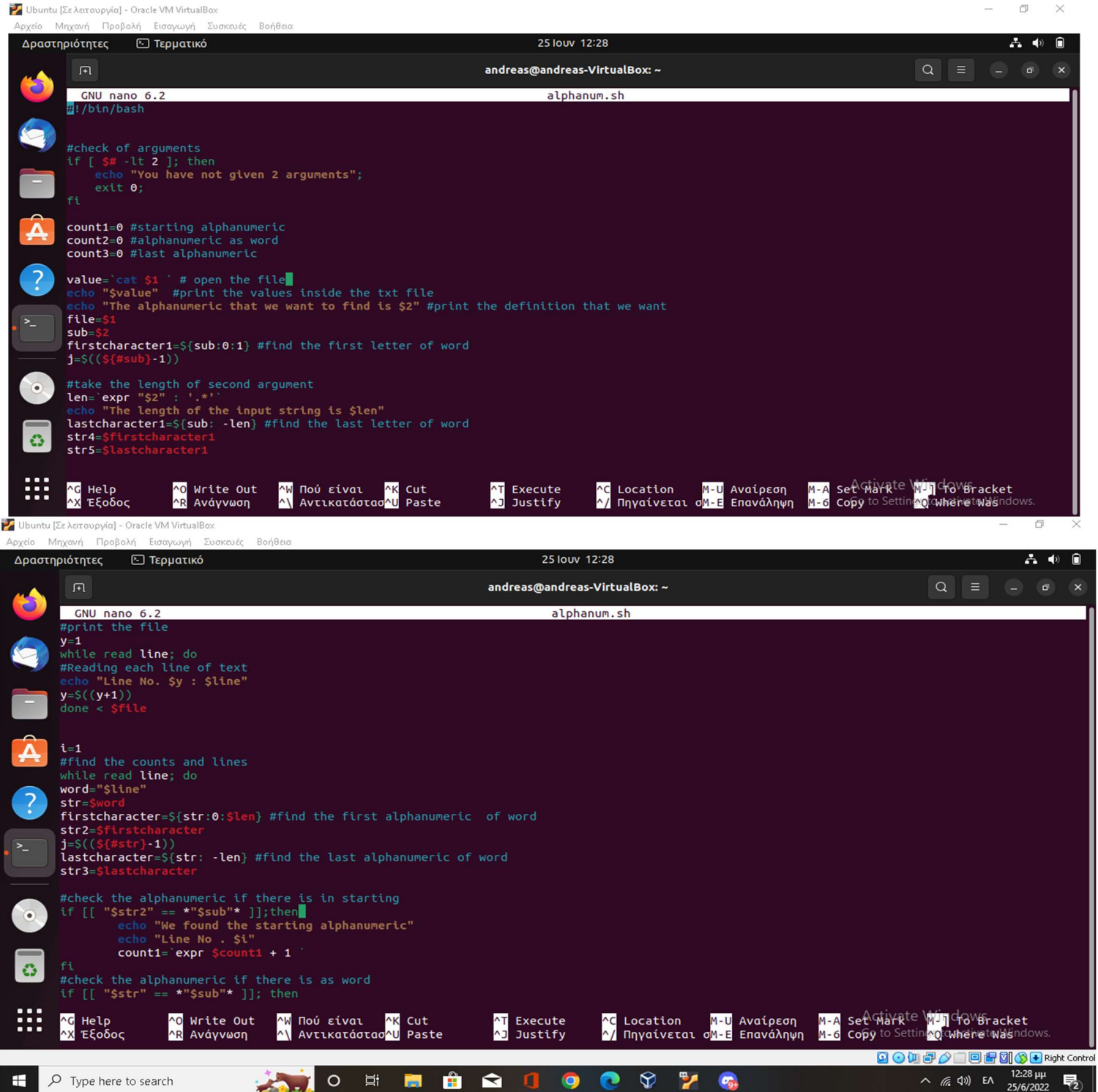


```
andreas@andreas-VirtualBox:~$ ./guess.sh
Two arguments required
andreas@andreas-VirtualBox:~$ ./guess.sh 1 -1
The second argument must be positive number
andreas@andreas-VirtualBox:~$

andreas@andreas-VirtualBox:~$ ./guess.sh 1 4
Fill the directory name (or E to exit) : business
Ypoerwtina 1 apotelesmata= 0
*****
Ypoerwtina 2 apotelesmata= 3
business/test1.txt
business/file3.txt
business/test2.txt
*****
Ypoerwtina 3 apotelesmata= 1
business
*****
Ypoerwtina 4 apotelesmata= 0
*****
Ypoerwtina 5 apotelesmata= 1
business/file3.txt
Fill the directory name (or E to exit) : Bivreo
Ypoerwtina 1 apotelesmata= 0
*****
Ypoerwtina 2 apotelesmata= 0
*****
Ypoerwtina 3 apotelesmata= 1
Bivreo
*****
Ypoerwtina 4 apotelesmata= 0
*****
Ypoerwtina 5 apotelesmata= 0
Fill the directory name (or E to exit) : folder
folder cannot be processed or is not folder
Fill the directory name (or E to exit) : E
-----Final Count-----
Ypoerwtina 1= 0
Ypoerwtina 2= 3
Ypoerwtina 3= 2
Ypoerwtina 4= 0
Ypoerwtina 5= 1
*****
Monopati Bivreo Ypoerwtina 1= 0, Ypoerwtina 2= 0, Ypoerwtina 3= 1, Ypoerwtina 4= 0, Ypoerwtina 5= 0
Monopati business Ypoerwtina 1= 0, Ypoerwtina 2= 3, Ypoerwtina 3= 1, Ypoerwtina 4= 0, Ypoerwtina 5= 1
```

## Άσκηση 3

### 1. Εμφάνιση του κώδικα με print screen:



The image shows two screenshots of a terminal window in Oracle VM VirtualBox, displaying the contents of a script named 'alphanum.sh' in nano 6.2 editor. The terminal window title is 'andreas@andreas-VirtualBox: ~'. The script is written in Bash and includes comments in Greek.

```
#!/bin/bash

#check of arguments
if [ $# -lt 2 ]; then
    echo "You have not given 2 arguments";
    exit 0;
fi

count1=0 #starting alphanumeric
count2=0 #alphanumeric as word
count3=0 #last alphanumeric

value=`cat $1` # open the file
echo "$value" #print the values inside the txt file
echo "The alphanumeric that we want to find is $2" #print the definition that we want
file=$1
sub=$2
firstcharacter1=${sub:0:1} #find the first letter of word
j=$(( ${#sub} - 1 ))

#take the length of second argument
len=`expr "$2" : '.*'`
echo "The length of the input string is $len"
lastcharacter1=${sub: -len} #find the last letter of word
str4=$firstcharacter1
str5=$lastcharacter1
```

The second screenshot shows the same script with additional logic for finding the first and last alphanumeric characters in each line of the input file. The script uses `while read line; do` loops to process each line and `if` statements to check for alphanumeric characters at the start and end of the line.

```
#print the file
y=1
while read line; do
    #Reading each line of text
    echo "Line No. $y : $line"
    y=$((y+1))
done < $file

i=1
#find the counts and lines
while read line; do
    word="$line"
    str=$word
    firstcharacter=${str:0:$len} #find the first alphanumeric of word
    str2=$firstcharacter
    j=$(( ${#str} - 1 ))
    lastcharacter=${str: -len} #find the last alphanumeric of word
    str3=$lastcharacter

    #check the alphanumeric if there is in starting
    if [[ "$str2" == *"$sub"* ]]; then
        echo "We found the starting alphanumeric"
        echo "Line No. $i"
        count1=`expr $count1 + 1`
    fi

    #check the alphanumeric if there is as word
    if [[ "$str" == *"$sub"* ]]; then
        count2=`expr $count2 + 1`
    fi
done
```

```
GNU nano 6.2 alphanum.sh
#check the alphanumeric if there is in starting
if [[ "$str2" == *"$sub"* ]];then
    echo "We found the starting alphanumeric"
    echo "Line No. $i"
    count1=`expr $count1 + 1 `
fi
#check the alphanumeric if there is as word
if [[ "$str" == *"$sub"* ]]; then
    echo "We found the alphanumeric as word"
    echo "Line No. $i"
    count2=`expr $count2 + 1 `
fi
#check the alphanumeric if there is at last
if [[ "$str3" == *"$sub"* ]]; then
    echo "We found the last alphanumeric"
    echo "Line No. $i"
    count3=`expr $count3 + 1 `
fi
i=$((i+1))
done < $file

#print the counts
echo "The count of starting alphanumeric is : $count1"
echo "The count of alphanumeric as word is : $count2"
echo "The count of alphanumeric at final is : $count3"
```

Help Write Out Πού είναι Αντικατάσταση Cut Execute Location M-U Αναίρεση M-A Set Mark M-o Copy to Setting no where was windows.

2. Πρώτα αρχικοποιούμε τις εξής τιμές: ένα πλήθος που θα μετράει πόσες φορές βρέθηκε το αλφαριθμητικό στην αρχή μιας λέξης, ένα δεύτερο πλήθος που θα μετράει πόσες φορές βρέθηκε το αλφαριθμητικό μέσα σε μία λέξη και τέλος ένα πλήθος που θα μετράει πόσες βρέθηκε το αλφαριθμητικό στο τέλος της λέξης. Όλα αυτά αφού πρώτα ελέγξουμε ότι έχουν δηλωθεί και τα δυο ορίσματα.
3. Έπειτα εμφανίζουμε το αρχείο το οποίο θέλουμε το οποίο το δηλώνουμε σαν όρισμα με την εντολή \$1 (στην προκειμένη περίπτωση είναι το test1.txt).
4. Τοποθετούμε και το αλφαριθμητικό(με την εντολή \$2 επειδή είναι το δεύτερο όρισμα που δηλώνουμε) σε μια μεταβλητή και βρίσκουμε το μήκος του το οποίο θα μας χρησιμεύσει αργότερα στο βήμα 7.
5. Ξαναεμφανίζουμε αναλυτικότερα το αρχείο με τις γραμμές του με την χρήση επαναληπτικής διαδικασίας
6. Στην συνέχεια χρησιμοποιώντας ξανά την ίδια επανάληψη βρίσκουμε τις γραμμές και τα πλήθη για το αλφαριθμητικό που ψάχνουμε.
7. Μέσα στην επανάληψη έχουμε την μεταβλητή str η οποία θα χρησιμεύσει στον έλεγχο για το αν υπάρχει το αλφαριθμητικό ως λέξη, την μεταβλητή firstcharacter από την οποία βρίσκουμε τους πρώτους χαρακτήρες του πίνακα για να γίνει η σύγκριση, την μεταβλητή str2 η οποία θα κρατάει το πρώτο αλφαριθμητικό από κάθε λέξη ώστε στην συνέχεια να ελέγχει αν αντιστοιχείται με το αλφαριθμητικό που έχουμε σαν όρισμα, ομοίως και η str3 στην οποία έχει αποθηκευτεί το lastcharacter μόνο που σε αυτή εκχωρείται το τελευταίο αλφαριθμητικό της λέξης. Κάθε φορά που μπαίνουμε σε μια από τις επιλογές

εμφανίζουμε την γραμμή που έχει βρεθεί το εκάστοτε αποτέλεσμα και αυξάνουμε το αντίστοιχο πλήθος κατά ένα.

8. Η πρώτη επιλογή συγκρίνει την `str2` η οποία έχει από κάθε γραμμή τα πρώτα αλφαριθμητικά όσα έχουν το ίδιο μήκος με της `sub` που έχουμε αποθηκεύσει το όρισμα για να βρίσκουμε αν το αλφαριθμητικό υπάρχει στην αρχή. Δεύτερη επιλογή εφόσον θέλουμε να βρούμε πόσες γραμμές περιέχουν το αλφαριθμητικό συγκρίνουμε απλά την `str` με την `sub`, αυτό σημαίνει ότι όταν βρίσκει είτε μια λέξη στην αρχή είτε στην μέση είτε στο τέλος θα την περιλαμβάνει και αυτή στην αύξηση του πλήθους. Τέλος συγκρίνουμε για να βρούμε στο τέλος το αλφαριθμητικό που ψάχνουμε την `str3` με την `sub` στην οποία `str3` έχουμε τα τελευταία αλφαριθμητικά ίδια σε μήκος με την `sub` και αν βρει ότι είναι ίδια αυξάνει το πλήθος.
9. Τέλος εμφανίζουμε τα πλήθη συνολικά εφόσον έχει ολοκληρωθεί η επανάληψη.

### Ενδεικτικά Αποτελέσματα:

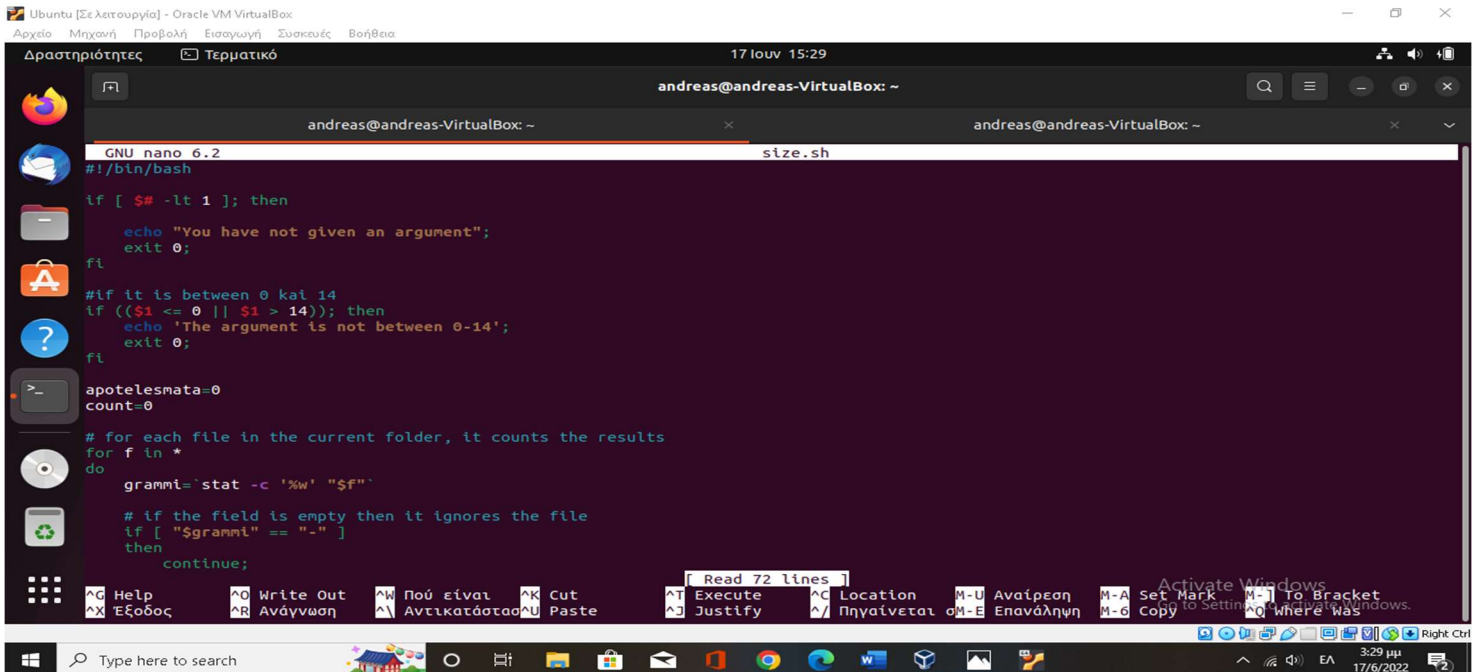
```
andreas@andreas-VirtualBox: ~  
andreas@andreas-VirtualBox:~$ nano alphanum.sh  
andreas@andreas-VirtualBox:~$ ./alphanum.sh test1.txt Find  
Find my phone  
Duckster team  
Cut the photo  
Close the window  
Open the terminal  
The alphanumeric that we want to find is Find  
The length of the input string is 4  
Line No. 1 : Find my phone  
Line No. 2 : Duckster team  
Line No. 3 : Cut the photo  
Line No. 4 : Close the window  
Line No. 5 : Open the terminal  
We found the starting alphanumeric  
Line No. 1  
We found the alphanumeric as word  
Line No. 1  
The count of starting alphanumeric is : 1  
The count of alphanumeric as word is : 1  
The count of alphanumeric at final is : 0  
andreas@andreas-VirtualBox:~$ ./alphanum.sh test1.txt the  
Find my phone  
Duckster team  
Cut the photo  
Close the window  
Open the terminal  
The alphanumeric that we want to find is the  
The length of the input string is 3  
Line No. 1 : Find my phone  
Line No. 2 : Duckster team  
Line No. 3 : Cut the photo  
Line No. 4 : Close the window  
andreas@andreas-VirtualBox:~$  
andreas@andreas-VirtualBox:~$ ./alphanum.sh test1.txt photo  
Find my phone  
Duckster team  
Cut the photo  
Close the window  
Open the terminal  
The alphanumeric that we want to find is photo  
The length of the input string is 5  
Line No. 1 : Find my phone  
Line No. 2 : Duckster team  
Line No. 3 : Cut the photo  
Line No. 4 : Close the window  
Line No. 5 : Open the terminal  
We found the alphanumeric as word  
Line No. 3  
We found the last alphanumeric  
Line No. 3  
The count of starting alphanumeric is : 0  
The count of alphanumeric as word is : 1  
The count of alphanumeric at final is : 1  
andreas@andreas-VirtualBox:~$
```



## Άσκηση 4:

**Παρατήρηση:** Σε αυτό το σημείο έγινε αναβάθμιση του λειτουργικού συστήματος από Ubuntu 20.04 σε Ubuntu 22.04 διότι η εντολή `stat` η οποία θα δούμε αργότερα στην παλιότερη έκδοση δεν μας έδινε την ώρα δημιουργίας του αρχείου από το πεδίο `birth`.

1. Εμφάνιση του κώδικα με `print screen`:



```
GNU nano 6.2 size.sh
#!/bin/bash

if [ $# -lt 1 ]; then
    echo "You have not given an argument";
    exit 0;
fi

# if it is between 0 and 14
if (($1 <= 0 || $1 > 14)); then
    echo "The argument is not between 0-14";
    exit 0;
fi

apotelesmata=0
count=0

# for each file in the current folder, it counts the results
for f in *
do
    grammi=`stat -c '%w' "$f"`

    # if the field is empty then it ignores the file
    if [ "$grammi" == "-" ]
    then
        continue;
    fi
done
```

Ubuntu [Ελεγχόμενη] - Oracle VM VirtualBox

Αρχείο Μηχανή Προβολή Εισαγωγή Συσκευές Βοήθεια

Δραστηριότητες Τερματικό 17 Ιουν 15:30

andreas@andreas-VirtualBox: ~

```
GNU nano 6.2 size.sh
else
#else, it then isolates the field with time
wra='echo $grammi | cut -d' ' -f2 | cut -d':' -f1'

# if the time ends with the time the file was created
let wra=wra+0

if [ "$1" -eq "$wra" ]
then
let apotelesmata++
fi
done

# if there are results
if (( apotelesmata > 0 ))
then
rm -f timefile
touch timefile

# for each file in the current folder it writes the file if it exists
for f in *
do
grammi='stat -c '%w' "$f"'

# if the field is empty then it ignores the file
```

Help Εξοδος Write Out Ανάγνωση Πού είναι Αντικατάσταση Cut Paste Execute Justify Location Πηγαίνεται σ M-U Αφαίρεση Επανάληψη M-A Set Mark M-1 To Bracket M-6 Copy

Δραστηριότητες Τερματικό 17 Ιουν 15:30

andreas@andreas-VirtualBox: ~

```
GNU nano 6.2 size.sh
for f in *
do
grammi='stat -c '%w' "$f"'

# if the field is empty then it ignores the file
if [ "$grammi" == "-" ]
then
continue;
else
#else, it then isolates the field with time
wra='echo $grammi | cut -d' ' -f2 | cut -d':' -f1'

# if the time ends with the time the file was created
let wra=wra+0
if [ "$1" -eq "$wra" ]
then
count='expr $count + 1'
echo $f >> timefile
fi
done
else
echo "There are no files created on time "$1
fi
echo "The number of files that create are: $count"
```

Help Εξοδος Write Out Ανάγνωση Πού είναι Αντικατάσταση Cut Paste Execute Justify Location Πηγαίνεται σ M-U Αφαίρεση Επανάληψη M-A Set Mark M-1 To Bracket M-6 Copy



2. Αρχικά, ελέγχουμε στον κώδικα μας αν έχει εισαχθεί το όρισμα που θέλουμε αλλιώς μας εμφανίζει κατάλληλο μήνυμα.
3. Αμέσως μετά γίνεται ο έλεγχος του ορίσματος το οποίο θα πρέπει να είναι μεγαλύτερο του 0 και μικρότερο ή ίσο με το 14 για να συνεχίσουμε στο επόμενο βήμα διαφορετικά μας εμφανίζει ένα μήνυμα και γίνεται έξοδος από το πρόγραμμα(Στην προκειμένη περίπτωση έχουμε γράψει ανάποδα την δομή ελέγχου ώστε αν είναι μικρότερο η ίσο του 0 ή μεγαλύτερο του 14 να εμφανίζει κατευθείαν το μήνυμα και να κάνει έξοδο).
4. Σε πρώτο στάδιο για κάθε αρχείο του τρέχοντος φακέλου μετράει τα αποτελέσματα. Επίσης μετράει πόσα αρχεία πληρούν τα κριτήρια της ώρας δημιουργίας το οποίο επιτυγχάνεται με την εντολή stat. Με την εντολή cut από την άλλη απομονώνω την στήλη της ώρας και ελέγχω αν ταιριάζει με την ώρα του ορίσματος και αν ταιριάζει τότε αυξάνεται ο μετρητής.
5. Σε δεύτερο στάδιο το οποίο θα εκτελεστεί σε περίπτωση που θα υπάρξουν αποτελέσματα γίνεται η διαδικασία που αναφέραμε στο 5<sup>ο</sup> βήμα με την μόνη διαφορά ότι τώρα το όνομα του αρχείου μας θα γράφεται στο timefile.

Ενδεικτικά αποτελέσματα:

Αρχικά εμφανίζουμε το script με λάθη(Εδώ το λάθος είναι ότι έχουμε εκχωρήσει το 12 στο οποίο δεν έχουμε κάποιο αρχείο και στην αρχή δεν έχουμε βάλει κάποιο όρισμα).Εν συνεχεία όταν καταχωρούμε ως όρισμα το 14 παρατηρούμε πως η διαδικασία δημιουργίας έχει επιτευχθεί 11 φορές που σημαίνει ότι στο αρχείο timefile θα έχω 11 αρχεία. Αυτό έχει επιτευχθεί μέσω της εντολής stat από την οποία χρειαζόμαστε το πεδίο birth που υποδηλώνει την ημερομηνία δημιουργίας ενός αρχείου.

```
andreas@andreas-VirtualBox:~$ ./size.sh
```

```
You have not given an argument
```

```
andreas@andreas-VirtualBox:~$ ./size.sh 12
```

```
There are no files created on time 12
```

```
The number of files that create are: 0
```

```
andreas@andreas-VirtualBox:~$ ./size.sh 14
```

```
The number of files that create are: 11
```

```
andreas@andreas-VirtualBox:~$ ls
```

```
fileforsize size.sh snap timefile
```

```
andreas@andreas-VirtualBox:~$ stat timefile
```

```
File: timefile
```

```
Size: 162          Blocks: 8          IO Block: 4096   κανονικό αρχείο
```

```
Device: 803h/2051d Inode: 526770      Links: 1
```

```
Access: (0664/-rw-rw-r-- )  Uid: ( 1000/ andreas)   Gid: ( 1000/ andreas)
```

```
Access: 2022-06-17 15:28:00.100231733 +0300
```

```
Modify: 2022-06-17 15:28:00.484233237 +0300
```

```
Change: 2022-06-17 15:28:00.484233237 +0300
```

```
Birth: 2022-06-17 15:28:00.100231733 +0300
```

```
andreas@andreas-VirtualBox:~$
```

Activate Windows

Go to Settings to activate Windows.