



ΧΑΡΟΚΟΠΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΜΑΤΙΚΗΣ

Γεώργιος-Νεκτάριος Εγγλέζος (it21824)

1^η Εργασία στο μάθημα **Λειτουργικά Συστήματα**

Καλλιθέα, 12 Δεκεμβρίου 2019

Περιεχόμενα	
Άσκηση 1	3
Κώδικας	3
Ενδεικτικές εκτελέσεις (screenshots):	4
Γενικά Σχόλια/Παρατηρήσεις	4
Με δυσκόλεψε / δεν υλοποίησα	4
Άσκηση 2	5
Κώδικας	5
Ενδεικτικές εκτελέσεις (screenshots):	6
Γενικά Σχόλια/Παρατηρήσεις	6
Με δυσκόλεψε / δεν υλοποίησα	6
Άσκηση 3	7
Κώδικας	7
Ενδεικτικές εκτελέσεις (screenshots):	8
Γενικά Σχόλια/Παρατηρήσεις	8
Με δυσκόλεψε / δεν υλοποίησα	8
Άσκηση 4	9
Κώδικας	9
Ενδεικτικές εκτελέσεις (screenshots):	10
Γενικά Σχόλια/Παρατηρήσεις	10
Με δυσκόλεψε / δεν υλοποίησα	10
Άσκηση 5	11
Κώδικας	11
Ενδεικτικές εκτελέσεις (screenshots):	12
Γενικά Σχόλια/Παρατηρήσεις	12
Με δυσκόλεψε / δεν υλοποίησα	12
Άσκηση 6	13
Κώδικας	13
Ενδεικτικές εκτελέσεις (screenshots):	14
Γενικά Σχόλια/Παρατηρήσεις	14
Με δυσκόλεψε / δεν υλοποίησα	14
Συνοπτικός Πίνακας	15

Άσκηση 1

Κώδικας

To shell script που δημιουργήθηκε μαζί με τα σχόλια είναι:

```
#!/bin/bash

find $HOME -type f -exec du -h {} + | sort -h -r | head
# find -type f = βρίσκει "αρχεία" στο Home directory και τα
subdirectories του
#du -h = disk usage in human readable (k,m ,g... )
#sort -h -r = human numeric sort and in reverse ( Δοκίμασα -nr numeric
sort αλλά δεν δούλεψε )
#head = τα πρώτα 10 αποτελέσματα
```

Ενδεικτικές εκτελέσεις (screenshots):

- Επιτυχής εκτέλεση script

```
geonek@linuxvm:~$ ./it21824_ex_1.sh
147M  /home/geonek/Downloads/VideoForTest.mp4
54M   /home/geonek/Downloads/code_1.40.2-1574694120_amd64.deb
35M   /home/geonek/Videos/Video4Test#2.mp4
15M   /home/geonek/Music/ImagineDragonsForTest.mp3
14M   /home/geonek/.config/google-chrome/pnacl/0.57.44.2492/_platform_specific/x86_64/pnacl_public_x86_64_pnacl_llc_nex
13M   /home/geonek/.config/google-chrome/Safe Browsing/UrlSoceng.store
8,9M  /home/geonek/.wine/drive_c/windows/syswow64/shell32.dll
8,9M  /home/geonek/.wine/drive_c/windows/system32/shell32.dll
7,2M  /home/geonek/.cache/mozilla/firefox/s8tyrpd.default-release/startupCache/scriptCache.bin
6,3M  /home/geonek/.config/google-chrome/Default/IndexedDB/https_drive.google.com_0.indexeddb.leveldb/000007.ldb
```

Παρατηρήσεις:

Στην εικόνα από πάνω την έτρεξα στον υπολογιστή μου ώστε να φανεί καλύτερα το αποτέλεσμα

Γενικά Σχόλια/Παρατηρήσεις

- Η **find -type f** βρίσκει “αρχεία” στο Home directory και τα subdirectories του .
- Η **du -h** σημαίνει disk usage in human readable (k,m ,g...) και μου εμφανίζει τον χώρο που πιάνουν τα αρχεία (και directories αν δεν έβαζα την find -type f).
- Η **exec** έχει τον ρόλο του να συνδυάσει τις δύο αυτές εντολές .
- Το **sort -h -r** ταξινομεί ανάποδα (από το μεγαλύτερο στο μικρότερο) τα αποτελέσματα της du.
- Και τέλος η **head** εμφανίζει τα πρώτα 10 αποτελέσματα.

Με δυσκόλεψε / δεν υλοποίησα

Αρχικά είχα βρει από το man της find αυτήν την εντολή :

(find \$HOME -type f -printf '%s %p\n' | sort -nr | head -10)

αλλά το αποτέλεσμα που εμφάνιζε παρόλο που ήταν σωστό δεν μπορούσα να βρω κάποια παράμετρο για την **-printf** για να κάνει τα byte σε human readable , οπότε μετά από αρκετή έρευνα στο διαδίκτυο και στο **man** της **find** βρήκα και χρησιμοποίησα την **exec** σε συνδυασμό με την **find** . Θα μπορούσα εννοείται με άλλες εντολές να πάρω την πρώτη στήλη και να κάνω τα μαθηματικά με το χέρι αλλά θα γινόταν πολύ περίπλοκη η λύση μου χωρίς να υπάρχει λόγος.

Άσκηση 2

Κώδικας

To shell script που δημιουργήθηκε μαζί με τα σχόλια είναι:

```
#!/bin/bash
echo "Enter Your user's name";
read name;#User's name

echo $name > Howmany
Words=$(wc -w Howmany |tr " " . | cut -d. -f1) #Πόσες λέξεις δόθηκαν

while (( $Words != 1 )) #Ελεγχος παραμέτρων
do
echo "Wrong input you can only insert one word"
read name
echo $name > Howmany
Words=$(wc -w Howmany |tr " " . | cut -d. -f1)
echo $Words
done
rm Howmany #Σβήνω το αρχείο που φτιάχτηκε για την wc

RESULT=$(grep -c '^$name' /etc/passwd) #Δίνει το πλήθος των γραμμών με
την λέξη στην μεταβλητή name
if [ "$RESULT" = "1" ];
#Ελέγχει αν το όνομα υπάρχει στο /etc/passwd
then
echo "The User:$(grep '^$name:' /etc/passwd | cut -d: -f1) is local"
# Λέει ότι ο χρήστης είναι τοπικός
```

```
echo "User ID info : $(grep '^$name': /etc/passwd | cut -d: -f5)"
#Εκτυπώνει μόνο την πέμπτη στήλη που αποθηκευμένο το user info
else
getent passwd | grep '^$name' > /dev/null      # Για να πάρω το exit
status ώστε να ελέγξω αν το όνομα βρέθηκε ή όχι στην λίστα των
απομακρυσμένων χρηστών
status=$?
    if [ $status = 0 ] # 0 = Exists 1-255 = Doesn't exist
    then
        echo "The User:$(getent passwd |grep '^$name'| cut -d: -f1)
isn't local" #ομοίως με πάνω
        echo "User ID info : $(getent passwd |grep '^$name'| cut -d:
-f5)"      #ομοίως με πάνω

    else
        echo "User $name doesn't exist"
    fi
fi
```


Ενδεικτικές εκτελέσεις (screenshots):

- Επιτυχής εκτέλεση script

```
Enter Your user's name
root
The User:root is local
User ID info : root
it21824@debian:~$ ./it21824_ex_2.sh
Enter Your user's name
it21824
The User:it21824 isn't local
User ID info : ΓΕΩΡΓΙΟΣ-ΝΕΚΤΑΡΙΟΣ ΕΓΓΛΕΖΟΣ
it21824@debian:~$ ./it21824_ex_2.sh
Enter Your user's name
it2183355
User it2183355 doesn't exist
it21824@debian:~$
```

Παρατηρήσεις:

Στην εικόνα από πάνω γίνονται τρεις διαφορετικές εκτελέσεις :

1. Η πρώτη εκτέλεση γίνεται με είσοδο το “root” που είναι τοπικός χρήστης .
2. Η δεύτερη εκτέλεση γίνεται με είσοδο το όνομα μου “it21824” που είναι απομακρυσμένος χρήστης .
3. Η τρίτη κλήση γίνεται με είσοδο κάποιο it που δεν υπάρχει .

- Έλεγχος παραμέτρων χρήστη

```
it21824@debian:~$ ./it21824_ex_2.sh
Enter Your user's name
it21824 it218255
Wrong input you can only insert one word
it21824
The User:it21824 isn't local
User ID info : ΓΕΩΡΓΙΟΣ-ΝΕΚΤΑΡΙΟΣ ΕΓΓΛΕΖΟΣ
it21824@debian:~$
```

Παρατηρήσεις:

Ο έλεγχος παραμέτρων για την είσοδο που δίνει ο χρήστης ελέγχει πόσες λέξεις έδωσε . Στην περίπτωση που δεν έδωσε καμία είσοδο ή έδωσε πάνω από μία λέξη τον ξαναρωτάει . Για τον σκοπό αυτό χρησιμοποίησα την εντολή **wc** . Κάνοντας ανακατεύθυνση εξόδου της μεταβλητής που διάβασα σε ένα αρχείο **Output**. Χρησιμοποιώ την εντολή **wc -w** που μου δίνει το πλήθος των λέξεων στο αρχείο . Στην συνέχεια επειδή το αποτέλεσμα της **wc** ήταν της μορφής “**2 Output**” χρησιμοποίησα την εντολή **wc -w Howmany |tr " " . | cut -d. -f1** μετατρέποντας το κενό σε “.” και πήρα την πρώτη σύλη με την **cut** που ήταν το πλήθος των λέξεων . Τέλος έβαλα αυτό σε μια **while** ώστε να επαναλαμβάνεται μέχρι σώτου ο χρήστης δώσει σωστό input .

(Αφού έκανα την Άσκηση 3 σκέφτηκα ότι θα μπορούσα να είχα βάλει απλά την εντολή **tr -d -c 0-9**)

Γενικά Σχόλια/Παρατηρήσεις

Ανάλυση κώδικα : Αρχικά για να είναι πιο κατανοητό ως προς τον χρήστη το τι input θα δώσει εκτυπώνω με την **echo** μια ερώτηση για είσοδο και με την **read** την αποθηκεύω σε μια μεταβλητή .

- Το πρώτο σκέλος του script είναι ο έλεγχος για το αν το username που δίνει ο χρήστης είναι τοπικός χρήστης (local user). Για αυτό τον σκοπό χρησιμοποίησα την εντολή **grep** ψάχνοντας το name στην λίστα **/etc/passwd** . Για να ελέγξω ότι ο χρήστης υπάρχει έχω την εντολή **RESULT=\$(grep -c '^\$name' /etc/passwd)** που κρατάει στην μεταβλητή **RESULT** το πλήθος των γραμμών που περιέχουν το

\$name . Αν το **RESULT** είναι μεγαλύτερου του “0” τότε αυτό σημαίνει πως ο χρήστης υπάρχει και είναι τοπικός . Στην περίπτωση αυτή χρησιμοποιώντας την εντολή **cut -d: -f5** σε συνδυασμό με την **grep \$name /etc/passwd** εμφανίζω την πέμπτη στήλη που περιέχει το user info (Δίνω στην cut ως delimiter το “:” για να χωρίσω τις λέξεις).

- Στην περίπτωση που η μεταβλητή **Result** περιέχει το νούμερο **0** , χρησιμοποιώ την εντολή **getent passwd | grep \$name** για να ελέγξω αν ο χρήστης είναι απομακρυσμένος αποθηκεύοντας το **exit status** της εντολής σε μια μεταβλητή .
- Αν το **exit status** είναι ίσο με **0** σημαίνει ότι η εντολή έτρεξε με επιτυχία και στην περίπτωση αυτή εκτυπώνω το user info (πέμπτη στήλη)του **\$name**
- Αν το **exit status** είναι οτιδήποτε άλλο (1-255 αν δεν κάνω λάθος) σημαίνει ότι ο χρήστης δεν βρέθηκε ούτε στην λίστα των απομακρυσμένων χρηστών οπότε εμφανίζω και ανάλογο μήνυμα .

Με δυσκόλεψε / δεν υλοποίησα

Άσκηση 3

Κώδικας

To shell script που δημιουργήθηκε μαζί με τα σχόλια είναι:

```
#!/bin/bash

Total=$(du -sh $Home )          #du -ch | grep total ή du -s για bytes
Num=$(echo $Total | tr -d -c 0-9) #Δίχνει μόνο το νούμερο στην
μεταβλητή Total
Letter=$(echo $Total | sed 's/[^A-Z?]*//g') #Δίχνει μόνο το γράμμα στην
μεταβλητή Total
#Πιστεύω δεν έχει νόημα να κάνω έλεγχο για kb

if [ "$Letter" = "M" ]
then
    Percent=$(bc <<<"scale=2;$Num/50") #VAR=$(echo "scale=2;$Num/50") or
VAR=$(bc <<<"scale=2;$Num/50")
    Num=$(echo "scale=3;$Num/1000"|bc)
    echo "Using $Percent% of quota (0$Num/5.0 GB)"
elif [ "$Letter" = "G" ]
then

    Num=$(echo $Num | tr ", " .)
    Percent=$(bc <<<"scale=2;$Num/5")
    Percent=$(echo $Percent | cut -d. -f2 )

    HowMany=$(awk -F '[0-9]' '{print NF-1}' <<< "$Num") #βλέπω πόσα
νούμερα υπάρχουν μέσα στην μεταβλητή Num
    if [[ $HowMany = 2 ]]
    then
        Num1=${Num:0:1} #Σπάω το νούμερο ώστε να βάλω το κόμμα ανάμεσα
        Num2=${Num:1:2}
        Num=$Num1.$Num2
    fi
fi
```

```
echo "Using $Percent% of quota ($Num/5.0 GB)"
```

```
fi
```

```
#https://stackoverflow.com/questions/12722095/how-do-i-use-floating-point-division-in-bash
```

```
#https://unix.stackexchange.com/questions/434964/how-do-i-the-extract-the-first-digit-from-a-number-variable-in-a-bash-script
```

```
#https://www.unix.com/shell-programming-and-scripting/164835-bash-keep-only-certain-characters.html
```

Ενδεικτικές εκτελέσεις (screenshots):

- Επιτυχής εκτέλεση script

```
geonek@linuxvm:~$ ./it21824_ex_3.sh
Using 14.74% of quota (0.737/5.0 GB)
geonek@linuxvm:~$ ./it21824_ex_3.sh
Using 20.30% of quota (0.1015/5.0 GB)
geonek@linuxvm:~$ ./it21824_ex_3.sh
Using 22% of quota (1.1/5.0 GB)
```

Παρατηρήσεις:

- Στην πρώτης δύο περιπτώσεις είναι λιγότερο από 1 gb
- Στην τρίτη περίπτωση ξεπερνάει το 1 gb

Γενικά Σχόλια/Παρατηρήσεις

- Αρχικά αποθηκεύω σε μία μεταβλητή **Total** το συνολικό χώρο που πιάνει το **\$Home** directory χρησιμοποιώντας την εντολή **du -sh** που μου δίνει τον χώρο σε human readable .
- Στην συνέχεια με τις εντολές **echo \$Total | tr -d -c 0-9** αποθηκεύω σε μία μεταβλητή **Num** μόνο το νούμερο που υπάρχει στην μεταβλητή **Total**.
- Και με την εντολή **echo \$Total | sed 's/[^A-Z?]*//g'** κρατάω σε μια μεταβλητή **Letter** μόνο το γράμμα το υπάρχει στην μεταβλητή **Total** που δηλώνει τη μονάδα μέτρησης του μεγέθους (π.χ. M).
- Τέλος με μία **if** ελέγχω το γράμμα που είναι στην μεταβλητή **Letter** . Ανάλογα με το γράμμα κάνω τις διαιρέσεις για να βρω το ποσοστό % . Για τον σκοπό αυτό χρησιμοποίησα την εντολή **bc** καθώς τα linux από μόνα τους δεν υποστηρίζουν

floating point πράξεις π.χ. Η `bc <<<"scale=2;$Num/50` το scale είναι τα δεκαδικά ψηφία . Στην περίπτωση που το μέγεθος είναι G χωρίζω τα δύο ψηφία και βάζω ανάμεσα ένα κόμμα για να εμφανιστεί σωστά στον χρήστη .

(Δεν νομίζω να υπάρχει περίπτωση ο χώρος να ναι kb οπότε δεν το έβαλα ως επιλογή)

Με δυσκόλεψε / δεν υλοποίησα

Για κάποιο λόγο ο κώδικάς μου ενώ δουλεύει στον υπολογιστή μου και σε αλλωνών όταν τον τρέχω με ssh στην σχολή μπορεί να εμφανίσει κάποιο error .

Άσκηση 4

Κώδικας

Το shell script που δημιουργήθηκε μαζί με τα σχόλια είναι:

```
#!/bin/bash
if ! hash write 2>/dev/null
#https://scripter.co/check-if-a-command-exists-from-shell-script/
then
    echo "Write isn't installed in this PC"
else
    echo "Write is installed"
    read -p 'Username: ' name

    echo $name > Howmany
    Words=$(wc -w Howmany |tr " " . | cut -d. -f1) #Πόσες λέξεις δόθηκαν
    while (( $Words != 1 )) #Έλεγχος παραμέτρων
    do
        echo "Wrong input, you can only insert one word"
        read -p 'try again: ' name
```

```

    echo $name > Howmany
    Words=$(wc -w Howmany | tr " " . | cut -d. -f1)

done
rm Howmany #Σβήνω το αρχείο που φτιάχτηκε για την wc

fi

#Ελεγχος για το αν ο χρήστης υπάρχει
RESULT=$(grep -c '^$name' /etc/passwd) #Δίνει RESULT 1 αν ο χρήστης
υπάρχει και 0 αν δεν υπάρχει
if [ "$RESULT" = "1" ];
#Ελέγχει αν ο χρήστης υπάρχει ως τοπικός χρήστης .
then
echo "The User:$(grep '^$name:' /etc/passwd | cut -d: -f1) exists"
Exists=True
else
getent passwd | grep '^$name' > /dev/null # Για να πάρω το exit status
και να δω αν υπάρχει ο χρήστης
status=$?
    if [ $status = 0 ] # 0 = Exists 1-255 = Doesnt exist
    then
        echo "The User:$(getent passwd | grep '^$name'| cut -d: -f1)
exists" #Εκτυπώνει ότι ο χρήστης υπάρχει (αππομακρισμένος)
        Exists=True

    else
        echo "User $name doesn't exist"
        Exists=False
    fi
fi

#Αν ο χρήστης υπάρχει ελέγχει αν είναι συνδεδεμένος
if [ $Exists = True ]
then

```



```

RESULT=$(who | grep -c '^$name') #Ομοίως με πάνω
if [[ $RESULT > 0 ]];             #ελέγχω με την εντολή who ποιος
είναι συνδεδεμένος
then
    echo "The User: $name is online "      #Αν είναι online
    echo -n "Give message : "
    write $name > /dev/null
    message=$?

    if [ $message = 0 ];
    then
        echo "message sent!"
    else
        echo "couldn't reach the user!"
    fi
else                                     #Αν δεν είναι online
    echo "The User isn't available right now"
fi
fi

```

Ενδεικτικές εκτελέσεις (screenshots):

- Επιτυχής εκτέλεση script

```
it21824@debian:~$ ./it21824_ex_4.sh
Write is installed
Username: it21824
The User:it21824 exists
The User: it21824 is online
Give message :Hello other terminal
message sent!
it21824@debian:~$
```

```
it21824@debian:~$
Message from it21824@debian on pts/49 at 17:43 ...
Hello other terminal
EOF

```

Παρατηρήσεις:

Εδώ έστειλα μήνυμα σε εμένα από το ένα terminal στο άλλο . Έλεγα αν υπάρχει ο χρήστης αν είναι online και στην συνέχεια μου έστειλα μήνυμα. Όταν σταμάτησα την write με Ctrl+D μου εμφανισε message sent.

- Έλεγχος παραμέτρων χρήστη

```
it21824@debian:~$ ./it21824_ex_4.sh
Write is installed
Username: it it
Wrong input, you can only insert one word
try again: it21824
The User:it21824 exists
The User: it21824 is online
Give message :

```

Παρατηρήσεις:

Για τον έλεγχο παραμέτρων χρησιμοποίησα την ίδια μεθοδολογία με την άσκηση 2 . (wc -c Output και μια while)

Γενικά Σχόλια/Παρατηρήσεις

- Για την άσκηση αυτή αξιοποίησα τον **κώδικα της άσκησης δύο** ώστε να ελέγξω για το αν υπάρχει ο χρήστης είτε ως τοπικός είτε ως απομακρυσμένος αποθηκεύοντας σε μια μεταβλητή **Exists** την λέξη **true** αν υπάρχει
- Στην περίπτωση που υπάρχει πηγαίνω και ελέγχω με την εντολή **who | grep -c '^\$name'** αν ο χρήστης είναι συνδεδεμένος αυτή την στιγμή .
- Στην περίπτωση που είναι συνδεδεμένος χρησιμοποιώ την εντολή **write** για να του στείλω μήνυμα αποθηκεύοντας το **exit status** της ώστε να ελέγξω αν η εντολή λειτούργησε σωστά και να εμφανίσω ανάλογο μήνυμα (απαραίτητο είναι ο χρήστης να σταματήσει την εκτέλεση της **write** με Ctrl+D)
- Στην περίπτωση που χρήστης δεν υπάρχει ή δεν είναι συνδεδεμένος εμφανίζει ανάλογο μήνυμα και τερματίζει την λειτουργία του προγράμματος .

Με δυσκόλεψε / δεν υλοποίησα

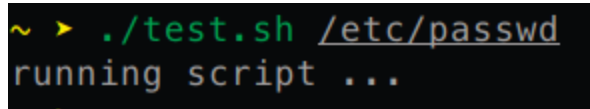
Άσκηση 5

Κώδικας

Το shell script που δημιουργήθηκε μαζί με τα σχόλια είναι:

Ενδεικτικές εκτελέσεις (screenshots):

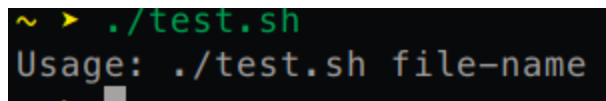
- Επιτυχής εκτέλεση script



```
~ > ./test.sh /etc/passwd  
running script ...
```

Παρατηρήσεις:

- Έλεγχος παραμέτρων χρήστη



```
~ > ./test.sh  
Usage: ./test.sh file-name
```

Παρατηρήσεις:

Γενικά Σχόλια/Παρατηρήσεις

Με δυσκόλεψε / δεν υλοποίησα

Άσκηση 6

Κώδικας

To shell script που δημιουργήθηκε μαζί με τα σχόλια είναι:

```
#!/bin/bash

##### INSERT TASK #####
InsertTask() {

echo "Insert a task:"
read TaskName
echo "Would you like to give a future date ? (yes or no )"
read answer
echo $answer > Howmany
Words=$(wc -w Howmany |tr " " . | cut -d. -f1) #Πόσες λέξεις δόθηκαν

while (( $Words != 1 )) || [[ "$answer" != "yes" && "$answer" != "no"
]] #Ελεγχος παραμέτρων
do

read -p 'Wrong input try again :' answer
echo $answer > Howmany
Words=$(wc -w Howmany |tr " " . | cut -d. -f1)

done
rm Howmany #Σβήνω το αρχείο που φτιάχτηκε για την wc

case $answer in
"yes")
    echo "The finale Format will be DDMMYYYYHHMM"
```

```

    read -p 'Day: (Use numbers 1-31)' Day      ##### For the day
#####
#
echo $Day > Howmany
Words=$(wc -w Howmany |tr " " . | cut -d. -f1) #Πόσες λέξεις δόθηκαν

while [ $Day -gt 31 ] || [ $Day -lt 1 ] #Ελεγχος παραμέτρων
do
    echo "Wrong input, you can only insert numbers from 1-31"
    read -p 'Try again: ' Day
    echo $Day > Howmany
    Words=$(wc -w Howmany |tr " " . | cut -d. -f1)

done
if [ $Day -lt 10 ]
then
Day=0$Day
fi
rm Howmany #Σβήνω το αρχείο που φτιάχτηκε για την wc

    read -p 'Month: (Use numbers 1-12)' Month      #####For the month
#####
#####

echo $Month > Howmany
Words=$(wc -w Howmany |tr " " . | cut -d. -f1) #Πόσες λέξεις δόθηκαν

while [ $Month -gt 12 ] || [ $Month -lt 1 ] #Ελεγχος παραμέτρων
do
    echo "Wrong input, you can only insert numbers from 1-12"
    read -p 'Try again: ' Month
    echo $Month > Howmany
    Words=$(wc -w Howmany |tr " " . | cut -d. -f1)

```

```

done
    if [ $Month -lt 10 ]
    then
        Month=0$Month
    fi
    rm Howmany #Σβήνω το αρχείο που φτιάχτηκε για την wc

    read -p 'Year: ' Year          ##### For the year
#####
#####

    echo $Year > Howmany
    Words=$(wc -w Howmany |tr " " . | cut -d. -f1) #Πόσες λέξεις δόθηκαν

    while [ $Year -gt 9999 ] || [ $Year -lt 2019 ] #Ελεγχος παραμέτρων
    do
        echo "Wrong input, you can only insert numbers 2019-9999"
        read -p 'Try again: ' Year
        echo $Year > Howmany
        Words=$(wc -w Howmany |tr " " . | cut -d. -f1)

    done

    rm Howmany #Σβήνω το αρχείο που φτιάχτηκε για την wc

    read -p 'Hour: (Use 1-24)' Hour #####for the hour
#####
#####

    echo $Hour > Howmany
    Words=$(wc -w Howmany |tr " " . | cut -d. -f1) #Πόσες λέξεις δόθηκαν

    while [ $Hour -gt 24 ] || [ $Hour -lt 1 ] #Ελεγχος παραμέτρων
    do
        echo "Wrong input, you can only insert numbers 1-24"
    done

```



```

    read -p 'Try again: ' Hour
    echo $Hour > Howmany
    Words=$(wc -w Howmany |tr " " . | cut -d. -f1)

done
    if [ $Hour -lt 10 ]
then
Hour=0$Hour
fi
rm Howmany #Σβήνω το αρχείο που φτιάχτηκε για την wc

    read -p 'Min: (Use 1-60)' Min          #####For the minutes
#####
###

    echo $Min > Howmany
Words=$(wc -w Howmany |tr " " . | cut -d. -f1) #Πόσες λέξεις δόθηκαν

while [ $Min -gt 60 ] || [ $Min -lt 1 ] #Ελεγχος παραμέτρων
do
    echo "Wrong input, you can only insert numbers 1-60"
    read -p 'Try again: ' Min
    echo $Min > Howmany
    Words=$(wc -w Howmany |tr " " . | cut -d. -f1)

done
    if [ $Min -lt 10 ]
then
Min=0$Min
fi
rm Howmany #Σβήνω το αρχείο που φτιάχτηκε για την wc

Date=$(echo $Day$Month$Year$Hour$Min)
HiddenDate=$(echo $Year$Month$Day$Hour$Min)

```

```

;;
"no")
    Date=$(date +%d%m%Y%H%M) #Μεταβλητή που κρατάει την ημερομηνία σε
    #Μεταβλητή που θα χρειαστεί στο
    HiddenDate=$(date +%Y%m%d%H%M)
    preview tasks
;;
esac

#https://www.lifewire.com/display-date-time-using-linux-command-line-403
#2698
echo "$HiddenDate:$Date:$TaskName" >> todo.list #anakateuthinsh eisodou
#στο arxeio todo.list (krataei ta prohgoumena)
}

##### DELETE TASK #####
DeleteTask() {

FILE=todo.list # Έλεγχος για το αν υπάρχει το αρχείο
if [ ! -f "$FILE" ]; then

    echo "You can't run options 2-6 if you don't have any tasks "
    exit 1
fi

#https://linuxize.com/post/bash-check-if-file-exists/

cat todo.list | cut -d: -f2,3
echo "Give the date or a 'key word' of the task you want to be deleted
:"
read WantDeleted
sed -i "$WantDeleted/"d ./todo.list

```

```

}

##### MODIFY TASK #####

ModifyTask() {

FILE=todo.list
if [ ! -f "$FILE" ]; then

    echo "You can't run options 2-6 if you don't have any tasks "
    exit 1
fi

cat todo.list | cut -d: -f2,3
echo "give the date of the task you want to modify"
read WantModified
DateModify=$(grep $WantModified todo.list | cut -d: -f2)
HiddenDate=$(grep $WantModified todo.list | cut -d: -f1)
TaskModify=$(grep $WantModified todo.list | cut -d: -f3)

echo "Would you like to edit the Date , Task or Both ?"
read change
echo $change > Howmany
Words=$(wc -w Howmany |tr " " . | cut -d. -f1)

while (( $Words != 1 )) || [[ "$change" != "Date" && "$change" !=
"Task" && "$change" != "Both" ]] #Ελεγχος παραμέτρων
do

read -p 'Wrong input try again :' change

```

```

echo $change > Howmany
Words=$(wc -w Howmany |tr " " . | cut -d. -f1)

done
rm Howmany #Σβήνω το αρχείο που φτιάχτηκε για την wc

case $change in
    "Date") # Είναι ο ίδιος κώδικας με την Insert task απλά κρατάω το
    προηγούμενο task και διαγράφω την προηγούμενη μορφή του .

        read -p 'Day: (1-31)' Day ##### For the day
#####
#
        while [ $Day -gt 31 ] || [ $Day -lt 1 ] #Ελεγχος παραμέτρων
        do
            echo "Wrong input, you can only insert numbers 1-31"
            read -p 'Try again: ' Day

        done
        if [ $Day -lt 10 ]
        then
            Day=0$Day
        fi

        read -p 'Month: (1-12)' Month #####For the month
#####
#####

        while [ $Month -gt 12 ] || [ $Month -lt 1 ] #Ελεγχος παραμέτρων
        do
            echo "Wrong input, you can only insert numbers from 1-12"
            read -p 'Try again: ' Month

```

```

done
    if [ $Month -lt 10 ]
    then
    Month=0$Month
    fi

    read -p 'Year: ' Year          ##### For the year
#####
#####

    while [ $Year -gt 9999 ] || [ $Year -lt 2019 ] #Ελεγχος παραμέτρων
    do
        echo "Wrong input, you can only insert numbers from 2019 to 9999"
        read -p 'Try again: ' Year

    done

    read -p 'Hour: ' Hour #####for the hour
#####
#####

    while [ $Hour -gt 24 ] || [ $Hour -lt 1 ] #Ελεγχος παραμέτρων
    do
        echo "Wrong input, you can only insert numbers 1-24"
        read -p 'Try again: ' Hour

    done

    if [ $Hour -lt 10 ]
    then
    Hour=0$Hour
    fi

```

```

    read -p 'Min: ' Min          #####For the minutes
#####
###

while [ $Min -gt 60 ] || [ $Min -lt 1 ] #Ελεγχος παραμέτρων
do
    echo "Wrong input, you can only insert numbers 1-60"
    read -p 'Try again: ' Min
done
if [ $Min -lt 10 ]
then
Min=0$Min
fi

DateModify=$Day$Month$Year$Hour$Min
HiddenDate=$Year$Month$Day$Hour$Min

sed -i "$WantModified/"d ./todo.list
    echo "$HiddenDate:$DateModify:$TaskModify" >> todo.list
;;
"Task")
echo "insert new Task"
    read TaskModify
    sed -i "$WantModified/"d ./todo.list
    echo "$HiddenDate:$DateModify:$TaskModify" >> todo.list
;;
"Both")
    InsertTask #Αφού θέλω να αλλάξω και τα δύο απλά καλώ την Insert
Task και φτιάχνω καινούργιο task
    sed -i "$WantModified/"d ./todo.list
;;
esac

```

```

}

##### Search Task #####

SearchTask() {

FILE=todo.list
if [ ! -f "$FILE" ]; then

    echo "You can't run options 2-6 if you don't have any tasks "
    exit 1
fi

echo "Give the Task or the Date to view the Task"
read answer
grep $answer todo.list | cut -d: -f2,3
}

##### Preview Task #####

PreviewTasks() {

FILE=todo.list
if [ ! -f "$FILE" ]; then

    echo "You can't run options 2-6 if you don't have any tasks "
    exit 1
fi

```

```

sort todo.list > todosorted
cat todosorted | cut -d: -f2,3
rm todosorted
}

##### Preview Daily Tasks #####

PreviewDailyTasks() {

FILE=todo.list
if [ ! -f "$FILE" ]; then

    echo "You can't run options 2-6 if you don't have any tasks "
    exit 1
fi

cat todo.list | cut -d: -f2,3 > TCTD
read -p 'Give the Day: (1-31)  ' Day

while [ $Day -gt 31 ] || [ $Day -lt 1 ] #Ελεγχος παραμέτρων
do

    echo "Wrong input, you can only insert numbers from 1-31"
    read -p 'Try again: ' Day

done

if [ $Day -lt 10 ]
then
Day=0$Day
fi

```



```

read -p 'Month: (1-12)' Month #####For the month
#####
#####

while [ $Month -gt 12 ] || [ $Month -lt 1 ] #Ελεγχος παραμέτρων
do
    echo "Wrong input, you can only insert numbers from 1-12"
    read -p 'Try again: ' Month

done
if [ $Month -lt 10 ]
then
Month=0$Month
fi

read -p 'Give the Year: ' Year

while [ $Year -gt 9999 ] || [ $Year -lt 2019 ] #Ελεγχος παραμέτρων
do
    echo "Wrong input, you can only insert years 2019-9999"
    read -p 'Try again: ' Year

done
SeeDate=$Day$Month$Year
egrep "$SeeDate[0-9]{4}" TCTD #
rm TCTD
}

```

```

##### EXIT #####

Quit() {
echo "Program closed Succesfully "
exit 1
}

#####
#####
##### MAIN #####
i=1
while [ $i = 1 ] #Για να γίνεται συνέχεια μέχρι ο χρήστης να πατήσει 7
do
echo "1) insert "
echo "2) delete "
echo "3) edit "
echo "4) search "
echo "5) preview"
echo "6) daily"
echo "7) quit"
echo "What do you want to do?"
read answer ;

case "$answer" in          # Case που καλούνται τις συναρτήσεις
    1)
        InsertTask
        ;;
    2)
        DeleteTask
        ;;
    3)
        ModifyTask
        ;;

```

```
4)
SearchTask
;;
5)
PreviewTasks
;;
6)
PreviewDailyTasks
;;
7)
Quit
;;
*) echo "Invalid option"    # for wrong input
;;
esac

done
```

Ενδεικτικές εκτελέσεις (screenshots):

- Επιτυχής εκτέλεση script

Insert Task: Γράφω το task και επιλέγω αν θα δώσω μελλοντική ημερομηνία ή άμα θα πάρει την τωρινή.

```
geonek@linuxvm:~$ ./it21824_ex_6.sh
1) insert
2) delete
3) edit
4) search
5) preview
6) daily
7) quit
What do you want to do?
1
Insert a task:
Task number 1
Would you like to give a future date ? (yes or no )
no
```

Delete task: Δίνω την ημερομηνία ή οτιδήποτε σχετικό με το task για να το σβήσω .

```
1) insert
2) delete
3) edit
4) search
5) preview
6) daily
7) quit
What do you want to do?
2
141220191658:Task number 1
141220191701:Task number 2
201220191010:task number 3
251220201515:Eminem
050330001515:NF
250420221212:Imagine Dragons
Give the date or a 'key word' of the task you want to be deleted :
141220191658
```

Edit task :Δίνω την ημερομηνία του task που θέλω να αλλάξω και επιλέγω τι θέλω να τροποποιήσω. Στην συγκεκριμένη περίπτωση επέλεξα να αλλάξω και την ημερομηνία και το Task.

```
1) insert
2) delete
3) edit
4) search
5) preview
6) daily
7) quit
What do you want to do?
3
141220191701:Task number 2
201220191010:task number 3
251220201515:Eminem
050330001515:NF
250420221212:Imagine Dragons
give the date of the task you want to modify
201220191010
Would you like to edit the Date , Task or Both ?
Both
Insert a task:
Edited task
Would you like to give a future date ? (yes or no )
no
```

Search Task: Δίνω είτε κάποια ημερομηνία ή λέξη και δείχνει την γραμμή .

```
1) insert
2) delete
3) edit
4) search
5) preview
6) daily
7) quit
What do you want to do?
4
Give the Task or the Date to view the Task
NF
050330001515:NF
```

Preview Tasks: Δείχνει ταξινομημένα κατά χρόνο όλα tasks. (Υποθέτω ότι δεν μπορεί να δώσει παλιές ημερομηνίες γιατί δεν υπάρχει νόημα σε αυτό)

```
1) insert
2) delete
3) edit
4) search
5) preview
6) daily
7) quit
What do you want to do?
5
141220191701:Task number 2
141220191748:Edited task
251220201515:Eminem
250420221212:Imagine Dragons
050330001515:NF
```

Daily Tasks : Ρωτάει τον χρήστη για κάποια συγκεκριμένη ημερομηνία και εμφανίζει όλα τα task αυτής της ημερομηνίας.

```
1) insert
2) delete
3) edit
4) search
5) preview
6) daily
7) quit
What do you want to do?
6
Give the Day: (Give a number 1-31) 14
Month: (Give a number 1-12)12
Give the Year: 2019
141220191701:Task number 2
141220191748:Edited task
```

```
1) insert
2) delete
3) edit
4) search
5) preview
6) daily
7) quit
What do you want to do?
7
Program closed Succesfully
geonek@linuxvm:~$
```

Παρατηρήσεις:

- Έλεγχος παραμέτρων χρήστη

```
1) insert
2) delete
3) edit
4) search
5) preview
6) daily
7) quit
What do you want to do?
2
You can't run options 2-6 if you don't have any tasks
```

Το μήνυμα που εμφανίζεται αν ο χρήστης επιλέξει 2-6 ενώ δεν υπάρχουν task

```
1) insert
2) delete
3) edit
4) search
5) preview
6) daily
7) quit
What do you want to do?
1
Insert a task:
I will check the parameters
Would you like to give a future date ? (yes or no )
no no
Wrong input try again :yees
Wrong input try again :yes
The finale Format will be DDMMYYYYHHMM
Day: (Use numbers 1-31)0
Wrong input, you can only insert numbers from 1-31
Try again: 28
Month: (Use numbers 1-12)13
Wrong input, you can only insert numbers from 1-12
Try again: 5
Year: 2000
Wrong input, you can only insert numbers 2019-9999
Try again: 2020
Hour: (Use 1-24)25
Wrong input, you can only insert numbers 1-24
Try again: 23
Min: (Use 1-60)0
Wrong input, you can only insert numbers 1-60
Try again: 59
```

Παρατηρήσεις:

Ο έλεγχος παραμέτρων που πραγματοποιήσα εδώ είναι αρχικά για την απάντηση στην ερώτηση αν θα δημιουργηθεί αυτόματη ημερομηνία ή αμα θα βάλει ο χρήστης δικιά του και στην συνέχεια για όλα τα νούμερα να μην ξεπερνούν τα φυσιολογικά πλαίσια. Παρόμοιος έλεγχος υπάρχει όπου ρωτάει κείμενο ή ημερομηνία . Χρησιμοποίησα την **wc** και την **while** .

Γενικά Σχόλια/Παρατηρήσεις

- Αρχικά έκανα το menu με την χρήση του case . Ανάλογα με το input του χρήστη (1-7) καλεί την ανάλογη συνάρτηση . Αν ο χρήστης δώσει άκυρο νούμερο τότε το πρόγραμμα τερματίζει . Αν το αρχείο todo.list δεν υπάρχει δεν θα τρέξουν οι επιλογές 2-6 (screenshot από πάνω).
- Insert Task : Αρχικά ζητάει από τον χρήστη να εισάγει ένα task που αποθηκεύεται σε μια μεταβλητή **TaskName** . Στην συνέχεια ρωτάει τον χρήστη αν το task του θα πάρει Ημερομηνία με την εντολή **Date** ή άμα θα δώσει εκείνος κάποια μελλοντική Ημερομηνία κάνοντας τους κατάλληλους ελέγχους εγκυρότητας για αυτή την περίπτωση . Η ημερομηνία αποθηκεύεται με δύο μορφές την κλασική HHMMEEEEΩΩΛΛ που θα φαίνεται στον χρήστη κατά την εκτέλεση του προγράμματος και μία κρυφή μορφή EEEEEMMHHΩΩΛΛ που θα χρησιμοποιηθεί αργότερα στην preview tasks και δεν φαίνεται πουθενά. Τέλος με ανακατεύθυνση εξόδου οι μεταβλητές αυτές αποθηκεύονται στο αρχείο **todo.list** (χρησιμοποιώ >> για να μην σβήνει τα προηγούμενα task).

(Εδώ έκανα **δύο υποθέσεις** πρώτα ότι οι μέρες πάνε μέχρι 31 και δεύτερο ότι ο χρήστης δεν θα δώσει παλιά ημερομηνία καθώς ελέγχω μόνο το έτος να είναι μεγαλύτερο ίσο του 2019)

- Delete Task : Παίρνοντας από τον χρήστη ως είσοδο κάποια ημερομηνία ή κάποια λέξη από το task που θέλει να σβήσει χρησιμοποιώ την εντολή **sed -i "\$WantDeleted/"d ./todo.list** για να σβήσω όλη την γραμμή και να αποθηκεύσω τις αλλαγές στο αρχείο **todo.list** (Για ευκολία του χρήστη εμφανίζω όλο το todo.list για να διαλέξει τι θέλει να σβήσει).
- Modify Task : Παρόμοια με πάνω εμφανίζω όλο το αρχείο και λέω στον χρήστη να επιλέξει κάποια ημερομηνία . Με το που δώσει ο χρήστης την μεταβλητή αυτή αποθηκεύω σε τρεις μεταβλητές το task, την ημερομηνία του task και την κρυφή του ημερομηνία με την βοήθεια της grep και της cut και σβήνω το task . Στην συνέχεια τον ρωτάω αν θέλει να αλλάξει το **Date** , **Task** ή και τα δύο **Both** και ανάλογα με το τι θα επιλέξει διαβάζω την ανάλογη μεταβλητή (κάνει τον έλεγχο παραμέτρων). Για παράδειγμα αν επιλέξει να αλλάξει το Date τον ρωτάω όπως και στο insert task ξεχωριστά ημέρα μήνα κλπ φτιάχνω τις δύο μεταβλητές για την ημερομηνία και τέλος κάνω τις τρεις μεταβλητές ξανά echo με ανακατεύθυνση εξόδου στο αρχείο . Στην περίπτωση που ο χρήστης επιλέξει το **both** καλείται η **συνάρτηση InsertTask** και στην ουσία γράφει ένα καινούργιο task .

- Search Task : Ο χρήστης δίνει την ημερομηνία ή κάποια λέξη ενός task και εμφανίζει το task με την βοήθεια της grep .
- Preview Tasks : Με την βοήθεια της κρυφής ημερομηνίας κάνω **sort** το **todo.list** με ανακατεύθυνση εξόδου σε ένα αρχείο **todosorted** . Στην συνέχεια από το ταξινομημένο αρχείο εμφανίζω την δεύτερη και τρίτη στήλη μόνο και τέλος με την εντολή **rm** σβήνω το αρχείο που φτιάχτηκε .
- Daily Tasks : Η εκφώνηση της άσκησης ήταν λίγο ασαφής οπότε έφτιαξα την συνάρτηση ώστε να ρωτάει τον χρήστη για ποια μέρα θέλει να εμφανιστούν τα tasks . Αρχικά δημιουργώ ένα αρχείο **TCTD** (to check todo.list) με την δεύτερη και τρίτη στήλη .Ο χρήστης δίνει Ημέρα, μήνα και έτος και δημιουργείται μια μεταβλητή της **SeeDate=\$Day\$Month\$Year** . Στην συνέχεια με την εντολή **egrep "\$SeeDate[0-9]{4}" TCTD** ταιριάζει όλα τα task με την ημερομηνία αυτή που έχουν την μορφή HHMMEEEExxxx όπου x είναι ένα τυχαίες ώρες που υπάρχουν . Τέλος με την **rm** σβήνω το αρχείο TCTD
- Quit : Εκτελεί την εντολή **Exit 1** και το πρόγραμμα τερματίζει .

Με δυσκόλεψε / δεν υλοποίησα

Συνοπτικός Πίνακας

1η Εργασία		
	Υλοποιήθηκε (ΝΑΙ/ΟΧΙ/ΜΕΡΙΚΩΣ)	Παρατηρήσεις
1η Άσκηση	ΝΑΙ	
2η Άσκηση	ΝΑΙ	
3η Άσκηση	ΝΑΙ	
4η Άσκηση	ΝΑΙ	
5η Άσκηση	ΟΧΙ	
6η Άσκηση	ΝΑΙ	