



ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ ΠΡΟΠΑΡΑΣΚΕΥΑΣΤΙΚΗ ΕΡΓΑΣΙΑ

ΠΑΠΑΝΔΡΙΚΟΠΟΥΛΟΣ ΓΡΗΓΟΡΙΟΣ-ΠΑΝΑΓΙΩΤΗΣ 03121136
ΝΤΑΒΕΑΣ ΣΤΑΣΙΝΟΣ 03121076

1) Προκειμένου να συνδεθούμε στο cslab κάναμε τις παρακάτω ενέργειες

```
C:\Users\GrigoriosPapandrik>ssh oslab025@orion.cslab.ntua.gr
The authenticity of host 'orion.cslab.ntua.gr (147.102.3.236)' can't be established.
ED25519 key fingerprint is SHA256:bA9q2rUKW1LvYf8uC1UaZTgL8DJiYW+NzvP7zrrsbdI.
This host key is known by the following other names/addresses:
  C:\Users\GrigoriosPapandrik/.ssh/known_hosts:3: orion.cslab.ece.ntua.gr
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'orion.cslab.ntua.gr' (ED25519) to the list of known host
s.
oslab025@orion.cslab.ntua.gr's password:
```

2) Έπειτα τυπώσαμε τον κατάλογο στον οποίο βρισκόμαστε μέσω της pwd

```
oslab025@os-node2:~$ pwd
/home/oslab/oslab025
oslab025@os-node2:~$
```

Και μέσω της ls -a εμφανίσαμε τα κρυφά αρχεία

```
oslab025@os-node2:~$ ls -a
. .bash_history .bashrc .bashrc.dpkg-dist .profile .vimrc
.. .bash_logout .bashrc_bak .config .ssh
oslab025@os-node2:~$
```

3) Στη συνέχεια μέσω της touch φτιάξαμε τα αρχεία που μας είχαν ζητηθεί test_file1, test_file2, test_file3 και μέσω της mkdir φτιάξαμε και τον ζητούμενο φάκελο test_dir

```
oslab025@os-node2:~$ touch test_file1 test_file2 test_file3
oslab025@os-node2:~$ mkdir test_dir
```

4)

Μέσω της εντολής mv μετονομάσαμε το αρχείο test_file3 στο όνομα GregoryStasinou

```
oslab025@os-node2:~$ mv test_file3 GregoryStasinou
```

5)

Έπειτα, μέσω της εντολής mv μεταφέραμε τα αρχεία test_file1 και test_file2 στο φάκελο test_dir

```
oslab025@os-node2:~$ mv test_file1 test_file2 test_dir
```

6) Στη συνέχεια, πηγαίμε στον φάκελο test_dir μέσω της cd και έπειτα τυπώσαμε σε χρονολογική σειρά τα περιεχόμενά του μέσω της ls -lt

```
oslab025@os-node2:~$ mv test_file1 test_file2 test_dir
oslab025@os-node2:~$ cd test_dir
oslab025@os-node2:~/test_dir$ ls -lt
total 0
-rw-r--r-- 1 oslab025 oslab 0 Feb 25 17:25 test_file1
-rw-r--r-- 1 oslab025 oslab 0 Feb 25 17:25 test_file2
oslab025@os-node2:~/test_dir$ |
```

7)

Με την rm διαγράψαμε την test_file2

```
oslab025@os-node2:~/test_dir$ rm test_file2
```

8) Με την cd .. βγήκαμε από την test_dir. Έπειτα, rmdir προσπαθήσαμε να διαγράψουμε την test_dir αλλά αυτό δεν ήταν εφικτό καθώς η rmdir διαγράφει φάκελο μόνο αν δεν περιέχει αρχεία

```
oslab025@os-node2:~/test_dir$ cd ..
oslab025@os-node2:~$ rmdir test_dir
rmdir: failed to remove 'test_dir': Directory not empty
```

Για αυτό στη συνέχεια χρησιμοποιήσαμε την rm -r προκειμένου να μπορέσουμε να διαγράψουμε τον φάκελο σωστά

```
oslab025@os-node2:~$ rm -r test_dir
```

9,10,11)

Έπειτα αντιγράψαμε το αρχείο file_generator.sh από τον κατάλογο /home/ oslab/code/shell-intro στον προσωπικό μας κατάλογο μέσω της cp. και εκτελέσαμε το αρχείο μέσω της εντολής ./file_generator.sh

Στη συνέχεια μέσω της grep -rn “oslab025” εντοπίσαμε όλες τις εμφανίσεις του username μας (oslab025) στον τρέχοντα κατάλογο καθώς και σε όλους τους υπο-καταλόγους, τυπώνοντας τον αριθμό γραμμής κάθε εμφάνισης της συμβολοσειράς σε κάθε αρχείο.

```
oslab025@os-node2:~$ cp /home/oslab/code/shell-intro/file_generator.sh ~/file_generator.sh
oslab025@os-node2:~$ ./file_generator.sh
oslab025@os-node2:~$ grep -rn "oslab025"
```

```
output-4.txt:36:oslab025
.bash_history:91:oslab025
output-2.txt:25:oslab025
output-2.txt:96:oslab025
test/output-1.txt:86:oslab025
test/output-5.txt:15:oslab025
test/output-5.txt:60:oslab025
test/output-5.txt:87:oslab025
test/output-9.txt:33:oslab025
test/output-9.txt:53:oslab025
test/output-6.txt:6:oslab025
test/output-6.txt:11:oslab025
test/output-6.txt:65:oslab025
test/output-7.txt:27:oslab025
test/output-7.txt:28:oslab025
test/output-3.txt:50:oslab025
test/output-3.txt:62:oslab025
test/output-3.txt:90:oslab025
test/output-8.txt:86:oslab025
test/output-10.txt:21:oslab025
```

12)Ανοίξαμε το output-7.txt μέσω της εντολής vim

```
oslab025@os-node2:~/test$ vim output-7.txt
```

και στην normal mode τρέξαμε την εντολή `%s/oslab025/gregorystasinos/g` αντικαθιστώντας το `oslab025` με το `gregorystasinos` η αλλαγή φαίνεται παρακάτω.

```
Entering Ex mode. Type "visual" to go to Normal mode.
M:%s/oslab025/gregorystasinos/g
```

```
oslab025@os-node2: ~/test
lsx0twa5BP
fwn2ser9sH
cKx39hjJGc
f5ZbspdWXF
uBxLP0yx24
nU0kImaQfC
qHnYQjHRi7
SDLJx3QJb8
J9UVTj39zs
b0KaLQVHH4
qQpKJxnYHi
CgCwvW0VKv
LjDzbEYZ2W
gregorystasinos
gregorystasinos
95800Citsp
U4y460Imyb
UAxdn0wxhj
eh0N7pbsrZ
```

ΑΣΚΗΣΗ 1.2

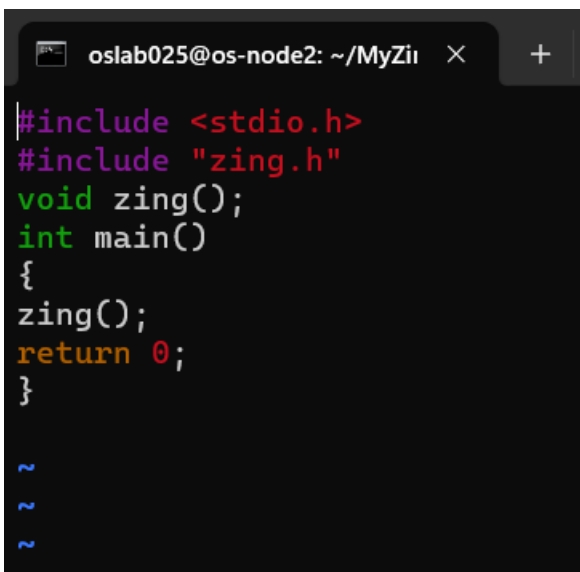
1) Η επικεφαλίδα “Header.h” στην γλώσσα c χρησιμοποιείται για την δήλωση συναρτήσεων και άλλων στοιχείων όπως μεταβλητές που πρόκειται να χρησιμοποιηθούν σε άλλα προγράμματα. Στην προκειμένη περίπτωση η zing.h περιέχει την δήλωση της συνάρτησης (και όχι την υλοποίηση της) zing().

2) Αρχικά, φτιάχνουμε ένα φάκελο MyZing και έπειτα αντιγράφουμε σε αυτόν τα αρχεία zing.h και Zing.c

```
oslab025@os-node2:~$ mkdir MyZing
oslab025@os-node2:~$ cp /home/oslab/code/zing/zing.h MyZing/
oslab025@os-node2:~$ cp /home/oslab/code/zing/zing.o MyZing/
oslab025@os-node2:~$ cd MyZing
oslab025@os-node2:~/MyZing$ ls
zing.h  zing.o
oslab025@os-node2:~/MyZing$
```

Έπειτα, υλοποιούμε την main.c η οποία το μόνο που κάνει είναι να καλεί την zing().

```
oslab025@os-node2:~/MyZing$ vim main.c
```



```
oslab025@os-node2: ~/MyZi  ×  +
#include <stdio.h>
#include "zing.h"
void zing();
int main()
{
    zing();
    return 0;
}

~
~
~
```

Έπειτα φτιάχνουμε την Makefile και στην αμέσως επόμενη εντολή μέσω της make εκτελούμε τις εντολές που περιέχει η Makefile

```
oslab025@os-node2:~/MyZing$ vim Makefile
oslab025@os-node2:~/MyZing$ make
gcc main.c zing.o -o main.o
```

Η makefile κάνει τις εξής διέργασίες:

- 1) Η main.o κάνει compile την main.c και δημιουργεί εκτελέσιμο αρχείο main.o.
- 2) Η zinglinking κάνει σύνδεση των αρχείων zing.o και main.o σε ένα εκτελέσιμο/executable αρχείο

```
oslab025@os-node2: ~/MyZin × + v
main.o: main.c zing.h
gcc main.c zing.o -o main.o
zinglinking: zing.o main.o
gcc zing.o main.o -o main
```

Τέλος, εκτελώντας το executable main.o μας εμφανίζεται στην οθόνη το ακόλουθο μήνυμα:

```
oslab025@os-node2:~/MyZing$ ./main.o
Hello, oslab025
```

3)

Αρχικά, δημιουργούμε ένα αρχείο zing2.c

```
oslab025@os-node2:~/MyZing$ vim zing2.c
```

Παρακάτω, παρουσιάζεται η υλοποίηση του zing2.c .Η getlogin() ανήκει στην βιβλιοθήκη unistd.h για αυτό την κάνουμε include. Η εντολή getlogin() επιστρέφει το name oslab25 στο bash το οποίο έπειτα τυπώνουμε μέσω της printf.

```
oslab025@os-node2: ~/MyZin × + v
#include "zing.h"
#include <stdio.h>
#include <unistd.h>

void zing()
{
    char *name;
    name = getlogin();
    printf("This is the login info: %s\n",name);
}
~
~
```

Παρακάτω, είναι η υλοποίηση του αλλαγμένου makefile:

```
oslab025@os-node2: ~/MyZii × + v
main.o: main.c zing.h
      gcc main.c zing.o -o main.o
      gcc main.c zing2.o -o main.o
zing2.o:zing2.c zing.h
      gcc -Wall -c zing2.c -o zing2.o

zinglinking1: zing.o main.c
      gcc zing.o main.c -o main1
zinglinking2: zing.o main.o
      gcc zing2.o main.c -o main2

clear:main1 main2 zing2.o
      rm main1 main2 zing.o
~
~
~
~
~
~
```

Η zing2.o κάνει compile την zing2.c και δημιουργεί το εκτελέσιμο zing2.o

Η main.o κάνει compile την main.c και δημιουργεί το εκτελέσιμο main.o και είτε για την περίπτωση που η συνάρτηση zing() έχει υλοποιηθεί από το zing2.c είτε από το zing.c

Τα zinglinking1 και zinglinking2 συνδέουν τα executable zing.o και main.o στο main1 και το zing2.o και main.o στο main2 αντίστοιχα.

Τέλος, η clear διαγράφει τα main1 και main2 zing.o που δημιουργήσαμε προηγουμένως.

Παρακάτω, εκτελούμε μέσω της make (label) μία μία την κάθε εντολή και τέλος τρέχουμε τα εκτελέσιμα main1 και main2 μέσω των εντολών ./main1 ./main2 Τα οποία τυπώνουν τα μηνύματα που εμφανίζονται στο cmd

```

oslab025@os-node2:~/MyZing$ vim Makefile
oslab025@os-node2:~/MyZing$ ls
main.c  main.o  Makefile  zing2.c  zing.h  zing.o
oslab025@os-node2:~/MyZing$ vim zing2.c
oslab025@os-node2:~/MyZing$ make zing2.o
gcc -Wall -c zing2.c -o zing2.o
oslab025@os-node2:~/MyZing$ make main.o
make: 'main.o' is up to date.
oslab025@os-node2:~/MyZing$ make zinglinking2
gcc zing2.o main.c -o main2
oslab025@os-node2:~/MyZing$ make zinglinking1
gcc zing.o main.c -o main1
oslab025@os-node2:~/MyZing$ ./main1
Hello, oslab025
oslab025@os-node2:~/MyZing$ ./main2
This is the login info: oslab025
oslab025@os-node2:~/MyZing$ |

```

4) Η makefile είναι χρήσιμη καθώς με μία μικρή αλλαγή που κάνουμε στο πρόγραμμα μπορούμε να τρέξουμε ξανά μόνο συγκεκριμένο κομμάτι του προγράμματος που μας ενδιαφέρει και όχι σε όλα από την αρχή χάρη στην τεχνική της διαχωρισμένης μεταγλώττισης (incremental compilation) μέσω της εντολής make (label).

5) Η εντολή που εκτελέσαμε δεν ήταν σωστή και είχε ως αποτέλεσμα το αρχείο foo.c να αντικατασταθεί από το εκτελέσιμο που παράχθηκε. Ειδικότερα, το gcc χρησιμοποιείται για να κάνει compile τον κώδικα. Η εντολή -Wall εκτυπώνει προειδοποιητικά κατά την μεταγλώττιση προκειμένου να αποφευχθούν όλα τα πιθανά προβλήματα του κώδικα. Η -o foo.c δηλώνει ότι το εκτελέσιμο που παράγεται θα έχει όνομα foo.c, και η foo.c είναι το αρχείο που θα μεταγλωττιστεί. Το λάθος προκλήθηκε από το γεγονός ότι το εκτελέσιμο αρχείο πήρε το ίδιο όνομα με το αρχείο που μεταγλωττίστηκε με αποτέλεσμα το εκτελέσιμο που δημιουργήθηκε να αντικαταστήσει το πηγαίο αρχείο.