ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ

Εργασία

Νικόλαος Κατωμέρης

ΑΕΜ: 8551

✉ ngkatomer@auth.gr

7^{Ο} ΕΞΑΜΗΝΟ
ΤΜΗΜΑ ΗΜΜΥ ΑΠΘ
ΙΑΝΟΥΑΡΙΟΣ 2018

Θέμα

Το θέμα της εργασίας ήταν η δημιουργία ενός απλού διερμηνέα γραμμής εντολών (command line interpreter) ή κελύφους (shell), το οποίο θα πρέπει να μπορεί να λειτουργήσει με δύο τρόπους:

- Interactive
- Batch

Παραδοτέο υλικό

Ζητήθηκε η παράδοση του πηγαίου κώδικα, ενός *makefile*, ενός αρχείου *README* με ορισμένες βασικές πληροφορίες τεκμηρίωσης του κώδικα και μιας αναφοράς που θα δείχνει πως λειτουργεί το πρόγραμμα.

Το παρόν έγγραφο (αναφορά) καθώς κι όλα τα υπόλοιπα ζητηθέντα αρχεία υπάρχουν στο github.

Μεταγλώττιση και εκτέλεση του προγράμματος

Η μεταγλώττιση του προγράμματος με χρήση του *makefile* μπορεί να γίνει ως εξής:

1. Μετάβαση στο φάκελο του πηγαίου κώδικα και του *Makefile*:
   - **cd** *path_to_source_folder*
2. Εκτέλεση μιας εκ των εντολών:
   - **make** (δημιουργείται το default πρόγραμμα **ntw_shell**)
   - **make fe_shell** (δημιουργείται το πρόγραμμα **fe_shell** (force-exit shell)
   - **make all** (δημιουργούνται και τα 2 προαναφερθέντα προγράμματα)

Η διαφορά του **ntw_shell** με το **fe_shell** είναι ότι αυτό **δεν** διακόπτει την εκτέλεσή του σε περίπτωση εισαγωγής λανθασμένης εντολής κατά το **interactive mode**.

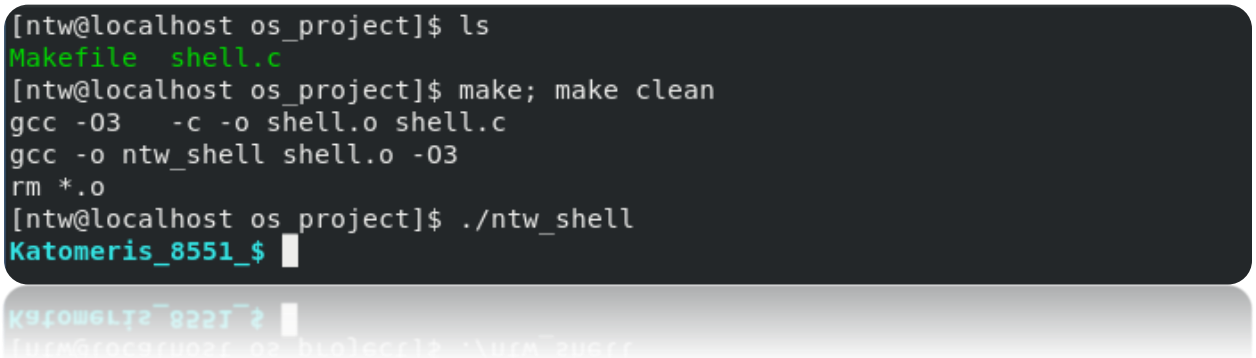
Η εκτέλεση του προγράμματος μπορεί να γίνει με τους εξής τρόπους:

1. **./ntw_shell** ή **./fe_shell** (interactive mode)
2. **./ntw_shell** *input_file* ή **./fe_shell** *input_file* (batch mode)

Έλεγχος προδιαγραφών-Παραδείγματα λειτουργίας του shell

Interactive mode:

✓ Τεστ 0: Μεταγλώττιση και εκτέλεση του προγράμματος σε interactive mode:



- ✓ Τεστ 1: Εκτέλεση μιας άδειας «εντολής», μίας απλής εντολής και μιας εντολής με παραμέτρους.

```
Katomeris_8551_$
Katomeris_8551_$
Katomeris_8551_$ who
ntw      tty2         2018-01-22 21:51 (/dev/tty2)
Katomeris_8551_$ date -d 1996-01-18 +%A
Thursday
Katomeris_8551_$
```

- ✓ Τεστ 2: Εκτέλεση περισσοτέρων εντολών διαχωρισμένων με τους χαρακτήρες «;» και «&&», με διάφορα τυχαία «κενά» (spaces) μεταξύ των εντολών και των χαρακτήρων ;/&&.

```
Katomeris_8551_$ who -b; date +%A;pwd;   ls
         system boot  2018-01-22 21:50
Tuesday
/mnt/hgfs/VM_Shared/os_project
Makefile  ntw_shell  shell.c
Katomeris_8551_$ date +%A&&pwd &&  who -b && ls
Tuesday
/mnt/hgfs/VM_Shared/os_project
         system boot  2018-01-22 21:50
Makefile  ntw_shell  shell.c
Katomeris_8551_$
```

Όπως φαίνεται στην παραπάνω εικόνα, οι εντολές εκτελέστηκαν σωστά και με τη σωστή σειρά και στις δύο περιπτώσεις.

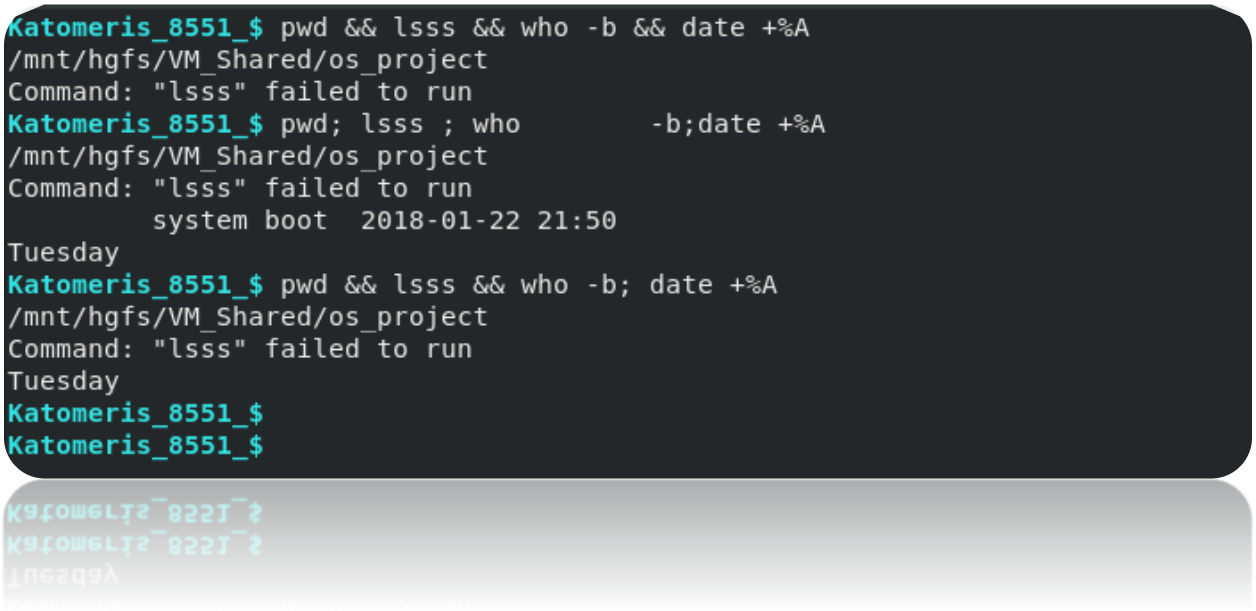
- ✓ Τεστ 3: Εκτέλεση ανύπαρκτων/με λανθασμένο τρόπο εντολών.

```
katomeris_8551_$ ls ./asdfasdfas
ls: cannot access './asdfasdfas': No such file or directory
Command: "ls" failed to run
Katomeris_8551_$ taime
Command: "taime" failed to run
Katomeris_8551_$ taime; pwd; who -b
Command: "taime" failed to run
/mnt/hgfs/VM_Shared/os_project
         system boot  2018-01-22 21:50
Katomeris_8551_$ taime && pwd && who -b
Command: "taime" failed to run
Katomeris_8551_$
```

Όπως φαίνεται στο στιγμιότυπο, η εκτέλεση μεμονωμένων ανύπαρκτων/με λανθασμένο τρόπο εντολών αποτυγχάνει.

Η εκτέλεση πολλών εντολών που χωρίζονται με «;» επιτυγχάνεται ακόμη κι αν κάποιες από τις εντολές δεν εκτελούνται, ενώ η εκτέλεσή τους όταν χωρίζονται με «&&» σταματά στην πρώτη αποτυχημένη εντολή.

Μερικά ακόμη, πιο περίπλοκα, τεστ/παραδείγματα φαίνονται στο παρακάτω στιγμιότυπο:



Όπως βλέπουμε, στην 3η περίπτωση η εντολή «**date +%A**» εκτελείται, αφού χωρίζεται από τις προηγούμενες με «**;**». Αντίθετα, στην 1η περίπτωση, λόγω του «**&&**», εφόσον δεν εκτελέστηκε η προηγούμενή της, δεν εκτελείται ούτε κι αυτή.

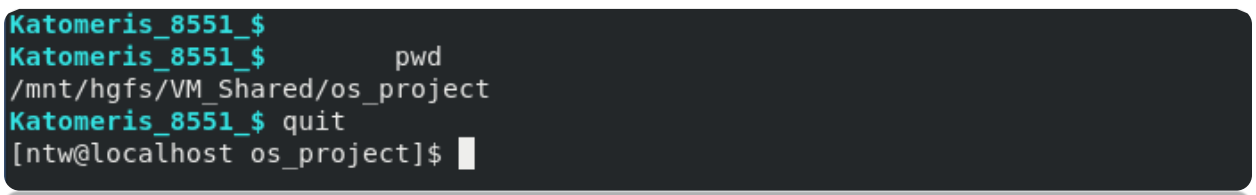
Στη 2η περίπτωση, εκτελούνται όλες οι εντολές που μπορούν να εκτελεστούν, γι' αυτές που δεν μπορούν εμφανίζεται -όπως σε κάθε τέτοια περίπτωση- το μήνυμα: «**Command: "…" failed to run**».

Σημειώνεται ότι:

Στις περιπτώσεις που παρουσιάστηκαν στο 3ο τεστ, αν αντί του **ntw_shell** εκτελούσαμε το **fe_shell**, η εκτέλεση του προγράμματος θα διακοπτόταν (με τη χρήση της κλήσης συστήματος «**exit()**») σε κάθε περίπτωση εισαγωγής λανθασμένης εντολής, στην οποία ΔΕΝ υπήρχε έπειτα ο χαρακτήρας «**;**».

✓ Τεστ 4: Εκτέλεση της «quit»

Η εντολή «**quit**» τερματίζει την εκτέλεση του προγράμματος.



Batch mode

Η εκτέλεση του προγράμματος με όρισμα κάποιο αρχείο (**batch mode**), θα οδηγήσει στην εκτέλεση όλων των εντολών που βρίσκονται στο αρχείο (γραμμή-γραμμή) με τον ίδιο ακριβώς τρόπο που θα εκτελούνταν αν εισάγονταν διαδοχικά στο **interactive mode**. Μερικά χαρακτηριστικά ειδικώς του **batch mode** είναι ότι:

- ✓ Το πρόγραμμα θα τερματίσει στο τέλος του αρχείου, είτε υπήρχε η εντολή «**quit**» είτε όχι.
- ✓ Το πρόγραμμα θα τερματίσει αμέσως στην περίπτωση που το αρχείο είναι κενό (ή περιέχει μόνο «κενά/tabs/newlines/etc») και θα εμφανίσει το μήνυμα: «***The specified file is empty***»

```
[ntw@localhost os_project]$ touch empty_file
[ntw@localhost os_project]$ ./ntw_shell empty_file
The specified file is empty.
[ntw@localhost os_project]$ echo "    " > empty_file
[ntw@localhost os_project]$ ./ntw_shell empty_file
The specified file is empty.
[ntw@localhost os_project]$
```

Batch mode test:

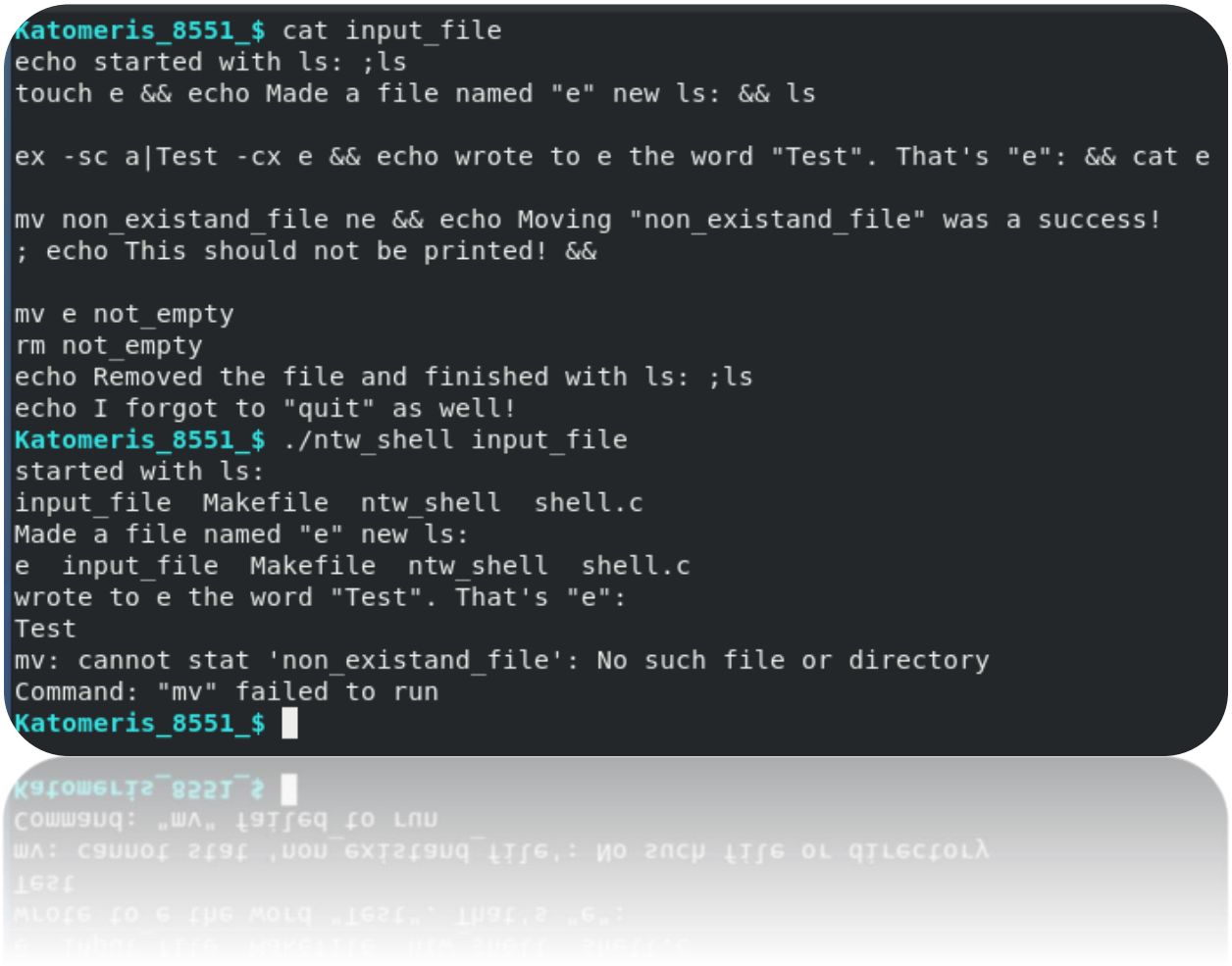
Δοκιμάστηκε το **batch mode** του προγράμματος τρέχοντάς το από το **interactive mode**:

```
Katomeris_8551_$ ls
input_file  Makefile  ntw_shell  shell.c
Katomeris_8551_$ cat input_file
echo started with ls: ;ls
touch e && echo Made a file named "e" new ls: && ls

ex -sc a|Test -cx e && echo wrote to e the word "Test". That's "e": && cat e

mv non_existand_file ne && echo Moving "non_existand_file" was a success! ;
; echo No success mv message was printed! &&

mv e not_empty
rm not_empty
echo Removed the file and finished with ls: ;ls
echo I forgot to "quit" as well!
Katomeris_8551_$ ./ntw_shell input_file
started with ls:
input_file  Makefile  ntw_shell  shell.c
Made a file named "e" new ls:
e  input_file  Makefile  ntw_shell  shell.c
wrote to e the word "Test". That's "e":
Test
mv: cannot stat 'non_existand_file': No such file or directory
Command: "mv" failed to run
No success mv message was printed!
Removed the file and finished with ls:
input_file  Makefile  ntw_shell  shell.c
I forgot to "quit" as well!
Katomeris_8551_$
```

Στο επάνω μέρος του στιγμιότυπου φαίνεται το αρχείο εισόδου και ύστερα η έξοδος του προγράμματος με όρισμα αυτό το αρχείο.

Αν, στο αρχείο ***input_file***, στην γραμμή με την εντολή «**mv**» δεν υπήρχε ο χαρακτήρας «**;**» στο τέλος, η εκτέλεση του προγράμματος θα σταματούσε στο σημείο αυτό, καθώς η εντολή αυτή αποτυγχάνει.

Αυτή η περίπτωση φαίνεται στο επόμενο στιγμιότυπο:



Επισήμανση:

Στην περίπτωση που υπάρχει λανθασμένη εντολή στο αρχείο εισόδου του **batch mode**, αν αντί του **ntw_shell** εκτελούσαμε το **fe_shell**, η εκτέλεση του προγράμματος θα διακοπτόταν (με τη χρήση της κλήσης συστήματος «**exit()**») ανεξαρτήτως της ύπαρξης ή όχι του χαρακτήρα «**;**» μετά την λανθασμένη εντολή.

Σε όλες τις περιπτώσεις, υποστηρίζονται ανά γραμμή εισόδου (είτε από αρχείο είτε όχι):

*Όλοι οι παρακάτω περιορισμοί μπορούν εύκολα να μεταβληθούν, αλλάζοντας τα **#defines** στον κώδικα.*

- Συνολικά έως και 512 χαρακτήρες (μαζί με όλους τους ειδικούς χαρακτήρες που περιλαμβάνει).
- Έως και 10 εντολές με έως και 10 ορίσματα ανά εντολή.
- Μέγεθος ενός ορίσματος έως και 420 χαρακτήρες.