# Αναφορά εργασίας Λειτουργικών Συστημάτων


## Άρης Ελευθέριος Παπαγγέλης

## ΑΕΜ: 8883


Το πρόγραμμά μας αποτελείται από ένα αρχείο πηγαίου κώδικα, το οποίο περιλαμβάνει τρείς συναρτήσεις.

Οι συναρτήσεις αυτές είναι οι εξής:

- int main (int argc, char *argv[])
- void shell (char buffer[])
- void execute (char *token, char *copy)

Παρακάτω ακολουθεί η ανάλυση της κάθε συνάρτησης.

## main

Η συνάρτηση αυτή είναι η κύρια συνάρτηση του προγράμματός μας, από την οποία και αρχίζει την εκτέλεση του. Δέχεται και ως όρισμα το όνομα του batchfile αρχείου από το command line, όταν πρόκειται για την batch λειτουργία.

Όταν δεν έχουμε δώσει κάποιο όνομα αρχείου ως όρισμα, η main θα εκτυπώνει το όνομα του prompt, θα μας ζητάει να δώσουμε μια γραμμή προς parsing και εκτέλεση και θα καλεί την συνάρτηση shell για να το επιτελέσει αυτό. Αυτή η διαδικασία θα γίνεται επαναληπτικά, μέχρι να δοθεί ως εντολή η λέξη quit, ή μέχρι να σταλεί στη διεργασία κάποιο σήμα διακοπής.

Όταν πάλι δώσουμε ως όρισμα όνομα αρχείου, τότε η main θα δημιουργήσει έναν file pointer προς το αρχείο, και θα συνεχίσει επαναληπτικά να διαβάζει γραμμές και να καλεί τη συνάρτηση shell. Η εκτέλεση θα σταματήσει είτε όταν τελειώσουν οι γραμμές του αρχείου, είτε όταν συναντηθεί η εντολή quit · όποιο από τα δύο επέλθει πρώτο.

## execute

Η συνάρτηση αυτή είναι η συνάρτηση που καλέιται από μια διεργασία –παιδί για την εκτέλεση της κάθε εντολής. Δέχεται σαν όρισμα την εντολή, token, και ένα αντίγραφο της αρχικής γραμμής που δόθηκε από το χρήστη, copy (αυτό χρειάζεται για να δούμε ποιός ήταν ο delimiter που χρησιμοποιήθηκε). Τα δύο ορίσματα δηλαδή είναι pointers σε χαρακτήρα (πίνακες char, δηλαδή strings).

Το αρχικό token θα σπάσει περεταίρω, με delimiters | > < , ώστε να δούμε αν υπάρχει input/output redirection ή pipes. Σίγουρα πάντως το πρώτο μέρος του αρχικού token θα

είναι η εντολή, με τυχόν παραμέτρους που αυτή μπορεί να έχει. Αυτό το πρώτο μέρος της εντολής το ονόμασα tok. Οι παράμετροι θα χωρίζονται από την εντολή με whitespace, επομένως θα γίνει ΑΚΟΜΑ ένα tokenization, του tok αυτή τη φορά, με whitespace ως delimiter. Αν η εντολή είναι quit, το παιδί θα τερματίσει με status 10. Σε διαφορετική περίπτωση, το όνομα της εντολής, καθώς και οι παράμετροι της, θα περαστούν σε κατάλληλους πίνακες, ώστε να δοθούν μετά σαν όρισμα στην execvp.

Ύστερα, θα πρέπει να διαχειριστούμε τα | > <, αν υπάρχουν. Έχουμε επομένως ένα loop, που αν εντοπιστούν τέτοιοι delimiters στην εντολή, θα απορρίψει καταρχάς χαρακτήρες whitespace που μπορεί να υπάρχουν στην αρχή ή το τέλος του substring, μιας και ουσιαστικά πρόκειται για ονόματα αρχείων, που πρέπει να είναι ακριβή, χωρίς whitespace.

Αν ο delimiter ήταν >, τότε έχουμε output redirection, επομένως ο file descriptor fdout θα δείχνει πια στο αρχείο που δόθηκε, αντί για το stdout που ήταν αρχικά.

Αν ο delimiter ήταν <, τότε έχουμε input redirection, επομένως ο file descriptor fdin θα δείχνει πια στο αρχείο που δόθηκε, αντί για το stdin που ήταν αρχικά.

Τέλος, αν ο delimiter ήταν |, τότε έχουμε pipe. Επομένως, βάζουμε τον file descriptor fdout να δείχνει σε ένα προσωρινό αρχείο, temp.txt, και θέτουμε την μεταβλητή –flag piping ίση με 1.

Η παραπάνω διαδικασία συνεχίζεται μέχρι να τελειώσουν οι delimiters τέτοιου τύπου. Επομένως, για > και <, θα κρατηθεί η πιο πρόσφατη εμφάνιση του delimiter. Δηλαδή, για παράδειγμα, η εντολή ls > a > b > c > d θα αποθηκεύσει στο αρχείο d.

Στη συνέχεια, με χρήση της εντολής dup2, θα ορίσουμε που θα γραφεί το αποτέλεσμα την εντολής που θα εκτελεστεί παρακάτω, με βάση και τα όσα αναφέρθηκαν προηγουμένως.

Ύστερα, υπάρχουν δύο ενδεχόμενα. Αν το flag piping τέθηκε ίσο με 1, τότε θα εκτλέσουμε την εντολή κανονικά, όμως το παιδί θα επιστρέψει με exit status 8, για να δείξουμε στη συνάρτηση shell ότι υπάρχουν pipes, τα οποία πρέπει να διαχειριστεί. Αν πάλι το piping ισούται με 0, θα εκτελεστεί απλά η εντολή, και θα επιστρέψει 0 για επιτυχή εκτέλεση.

Σε κάθε περίπτωση, ανεπιτυχής εκτέλεση εκτυπώνει κατάλληλο μήνυμα λάθους και επιστρέφει exit status 1.

Η εκτέλεση γίνεται με την εντολή execvp, η οποία δέχεται ορίσματα τα οποία περιγράφηκαν στην παραπάνω ανάλυση.

## shell

Η συνάρτηση αυτή είναι η συνάρτηση που κάνει το parsing των εντολών, και αποφασίζει για το αν υπάρχουν pipes ή όχι. Δέχεται ως όρισμα το buffer στο οποίο βρίσκεται η γραμμή που διαβάστηκε, είτε από το command line είτε από batchfile.

Η συνάρτηση ξεκινάει σπάζοντας την γραμμή σε tokens, με βάση τους delimiters ; & και \n (newline). Αν το πρώτο token δεν είναι κενό, τότε δημιουργεί μια διεργασία –παιδί η οποία

καλεί την συνάρτηση execute για την εκτέλεση της εντολής. Αν βέβαια ο delimiter δεν είναι διπλό & αλλά μονό, ή αν η εκτέλεση της προηγούμενης εντολής απέτυχε για delimiter &&, τότε η παρούσα εντολή δεν εκτελείται και τυπώνεται κατάλληλο μήνυμα λάθους. Οι έλεγχοι για προβληματικούς delimiters δηλαδή γίνονται μέσα στη διεργασία –παιδί. Σε κάθε περίπτωση, σε ανεπιτυχή εκτέλεση το παιδί επιστρέφει status 1, ενώ σε επιτυχή εκτέλεση θα επιστρέψει status 0.

Ύστερα, αφού το παιδί τερματίσει, θα πάρουμε το exit status του. Αν το exit status είναι 10, σημαίνει πως δόθηκε εντολή quit και το πρόγραμμά μας θα πρέπει να τερματίσει. Αν όμως το exit status ήταν 8, σημαίνει πως εντοπίστηκαν pipes στην εντολή που δόθηκε, τα οποία θα πρέπει να διαχειριστούμε. Αυτό γίνεται ως εξής:

Η διεργασία –παιδί παραπάνω εκτέλεσε μόνο την πρώτη εντολή του pipe, όπως μπορεί να καταλάβει κανείς βλέποντας τη συνάρτηση execute. Επομένως, θα κάνουμε advance το αρχικό token μας στην επόμενη εντολή του pipe.

Το αρχείο temp.txt που περιγράφηκε παραπάνω στην execute, θα αντιγραφεί σε αρχείο temp1.txt. Ύστερα, η επόμενη εντολή του pipe θα τροποποιηθεί, συνενώνοντας το string “<temp1.txt” στο τέλος της. Δηλαδή, ουσιαστικά το output της πρώτης εντολής καθίσταται input της δεύτερης, χρησιμοποιώντας το σύστημα αρχείων (καταλαβαίνω πως αυτή δεν είναι η βέλτιστη υλοποίηση, μια βελτίωση που θα μπορούσα να κάνω στο μέλλον είναι να το υλοποιήσω με την εντολή pipe για επικοινωνία μεταξύ γονέα –παιδιού).

Στη συνέχεια η τροποποιημένη εντολή θα δοθεί ως όρισμα στην execute, μέσω νέας διεργασίας παιδιού που δημιουργείται. Αν τυχόν υπάρχει και άλλο pipe (τριπλό, τετραπλό ή και παραπάνω), πάλι θα επιστραφεί exit status 8, και η παραπάνω διαδικασία θα επαναληφθεί, μέχρις ότου να τελειώσουν όλα τα pipes.

Τέλος, αφού έχει τελειώσει η διαδικασία για την παρούσα εντολή, θα προχωρήσουμε στην επόμενη εντολή, που χωριζόταν με ; ή &&.

Σε κάθε περίπτωση, η μνήμη που δεσμεύτηκε δυναμικά, μέσω κλήσεων strdup σε διάφορα σημεία, θα απελευθερωθεί στο τέλος της συνάρτησής μας.

## Επιπρόσθετες παρατηρήσεις

Όλες οι απαραίτητες βιβλιοθήκες για την κλήση των χρησιμοποιούμενων system calls έχουν συμπεριληφθεί στην αρχή του κώδικα με εντολές include.

Αναλυτικά σχόλια με περισσότερες τεχνικές λεπτομέρειες βρίσκονται σε κάθε γραμμή του πηγαίου κώδικα. Τα σχόλια στον κώδικα είναι στα Αγγλικά.

Η γραμματοσειρά που χρησιμοποιήθηκε στην εν λόγω αναφορά είναι Calibri, με μέγεθος 11. Σε περίπτωση που υπάρχει πρόβλημα, μπορείτε να επικοινωνήσετε μαζί μου να την αλλάξω.

Για οποιαδήποτε εκκρεμότητα ή ελλειψη μπορεί να υπάρχει, μπορείτε να με ενημερώσετε στην εξής διεύθυνση email:

aris.papagelis@gmail.com