ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ

Λειτουργικά Συστήματα Ι
2015-2016

Εργασία 1

Φώτιος Διονυσόπουλος ΑΜ : 5753
Email : dionys@ceid.upatras.gr

Εισαγωγικά

Σκοπός της εργασίας ήταν η κατασκευή ενός Shell.Το Shell είναι ένα πρόγραμμα το οποίο δέχεται (απο τον χρήστη) εντολές μέσω του πληκτρολογίου και τις εκτελεί μέσω διεργασιών.Πρέπει να σημειωθεί ο,τι ο κώδικας του 1^{ου} και 2^{ου} Shell είναι ακριβώς ο ίδιος διότι οι διαφορές μεταξύ τους είναι ελάχιστες.

Μέρος 1 & 2 (mysh1, mysh2)

Σκοπός του 1^{ου} και 2^{ου} μέρους ήταν η κατασκευή ενός Shell το οποίο θα διαβάζει μια είσοδο(το όνομα προγράμματος και τυχόν παραμέτρους) και θα εκτελεί το αντίστοιχο πρόγραμμα σε οποιοδήποτε directory.

Περιγραφή Λογικής:

Στη βασική συνάρτηση Main εκτελείται μόνο η συνάρτηση Shell() .

Η Shell() είναι η κύρια συνάρτηση στην οποία βρίσκεται ο While Loop.
Αρχικά, γίνεται δέσμευση για τον πίνακα των arguments χρησημοποιώντας την malloc ,στη συνέχεια με τη χρήση της getline παίρνουμε είσοδο απο τον χρήστη.Στο επόμενο μέρος, μέσω της συνάρτησης strtok “κομματιάζουμε” τη συμβολοσειρά με βάση τον κενό χαρακτήρα (" \t\r\n\a") εισόδου και περνάμε τα ορίσματα στον πίνακα arguments.Εν τω μεταξύ, αν και όπου χρειάζεται , γίνεται δυναμική δέσμευση μνήμης μέσω της realloc.

Το τελεύταίο μέρος της συνάρτησης Shell() είναι και το βασικότερο.
Ο πίνακας των ορισμάτων περνά ως όρισμα στη συνάρτηση η οποία είναι υπεύθυνη για την εκτέλεση των εντολών.Πρώτα, γίνεται έλεγχος εάν το πρώτο κελί του πίνακα είναι το “cd” ,εάν είναι τότε δεν γίνεται κανένα fork() και εκτελείται η εντολή chdir() αλλάζοντας έτσι , το $path.Εάν πρόκειται για κάποια άλλη εντολή τότε “κάνουμε fork ” τη διεργασία και δημιουργούμε δύο,η πρώτη (το παιδί) αναλαμβάνει την εκτέλεση του προγράμματος μέσω της execvp ,ενώ ο γονέας περιμένει μέχρι το παιδί να εκτελέσει το πρόγραμμα, μόλις εκτελεστεί η εντολή η διεργασία γονέας επιστρέφει στον While Loop.

Προβλήματα και λύσεις:

Το βασικότερο πρόβλημα που προέκυψε ήταν η επιλογή της κατάλληλης exec.Η exec επιτρέπει σε μια διεργασία να αλλάξει το πρόγραμμα που εκτελεί, αλλά παραμένοντας η ίδια διεργασία.Απο τις execl, execlp, execle, execv, execvp, execvpe επιλέχθηκε η execvp δίοτι τα arguments ηταν τοποθετημένα σε πίνακα χαρακτήρων και επειδή η execvp δεν χρειάζεται το absolute ή το relative path αλλά μπορεί απο μόνη της να βρεί την τοποθεσία του εκτελέσιμου.

Μέρος 3 (mysh3)

Σκοπός του 3ου μέρους ήταν η κατασκευή ενός Shell το οποίο θα διαβάζει μια είσοδο η οποία ενδέχεται να περιέχει το πολύ ένα pipe.

Περιγραφή Λογικής:

Αρχικά, όπως και προηγουμένως, γίνεται η δέσμευση της μνήμης για τον πίνακα των εντολών και στη συνεχεια ο χρηστης πληκτρολογεί την είσοδο στο shell.Μια πρώτη βασική διαφορά απο τα mysh1 και mysh2 είναι οτι στο mysh3(και στο mysh4) η είσοδος αρχικά "κομματιάζεται" στα pipes "|" και τοποθετείται στον πίνακα με τις εντολές.Στη συνέχεια η κάθε εντολή "κομματιάζεται" με βάση τον κενό χαρακτήρα .

Αφού ολοκληρωθεί η παραπάνω διαδικασία καλείται η συνάρτηση run η οποία αναλαμβάνει την εκτέλεση της εντολής.Η run,"κάνει fork" στη διεργασία, το παιδί εκτελεί την πρώτη εντολή και αντικαθιστά το stdout της με το Write end του pipe και στη συνέχεια γίνεται ξανά fork στη διεργασία του γονέας ,το νέο παιδί αναλαμβάνει την εκτέλεση του δεύτερου προγράμματος έχοντας πάρει ως είσοδο την έξοδο του προηγούμενου προγράμματος.Μόλις εκτελεστεί και το δεύτερο πρόγραμμα τα αποτελέσματα εμφανίζονται στην οθόνη και η διεργασία (γονέας) συνεχίζει την εκτέλεση του Shell.

Προβλήματα και λύσεις:

Ενα κύριο πρόβλημα ήταν ο τρόπος με τον οποίο θα γινόταν ο χειρισμός των εισόδων και εξόδων των εντολών και η εγγραφη/ανάγνωση απο τα pipes.Τη λύση έδωσε η dup2() η οποία βοήθησε να αντικατασταθεί το stdout ή το stdin με την εγγραφή/ανάγνωση στο file descriptor.

Μέρος 4 (mysh4)

Σκοπός του 4ου μέρους ήταν η κατασκευή ενός Shell το οποίο θα διαβάζει μια είσοδο η οποία ενδέχεται να περιέχει και πολλαπλά pipes.

Περιγραφή Λογικής:

Ομοίως με το mysh3, δεσμεύεται μνήμη και στη συνέχεια η είσοδος "κομματιάζεται" στα pipes ("|") και στη συνέχεια στα κενά.

Έπειτα, καλείται η συνάρτηση start_pipes η οποία μέσω ενός for Loop δημιουργεί ένα pipe() και στη συνέχεια καλέι τη συνάρτηση start_procs().Στη start_procs αρχικά, στην πρώτη εντολή και αντιγράφουμε το stdout στο fd[1](write end of the pipe).Την είσοδο της επόμενης προς εκτέλεση εντολής την λαμβάνουμε μέσω του pipe και αντικαθιστούμε το stdout της ξανά με το Write end του pipe.Η παραπάνω διαδικασία επαναλαμβάνεται μέχρις ότου φτάσουμε στην τελευταία εντολή οπού

εκτελείται(έξω απο τον for loop) με είσοδο την έξοδο της προ-τελευταίας εντολής και εμφανίζεται στην οθόνη το επιθυμιτό αποτέλεσμα.

Προβλήματα και λύσεις:

Ενα κύριο πρόβλημα ήταν ο τρόπος με τον οποίο θα γινόταν ο χειρισμός των εγγραφών/αναγνώσεων των πολλών προγραμμάτων στο file descriptor.Το πρόβλημα αυτό λύθηκε μέσω ενός for loop ο οποίος χρησημοποιούσε την struct (η οποία περιέχει τα ορίσματα του κάθε προγράμματος) και μέσω της start_procs με τη βοήθεια του dup2 έγιναν οι απαραίτητες αντικαταστάσεις του stdin/stdout απο και πρός το file descriptor.