Mysh1

Καθώς η υλοποίηση του mysh1 βασίζεται σχεδόν αποκλειστικά στην ανάγνωση εντολών απο το stdin, το κύριο πρόβλημα ήταν αυτό του parsing των εντολών. Για να επιτευχθεί αυτο δημιουργήθηκε η συνάρτηση parser.

Η αρχική ιδέα ήταν η συνάρτηση parser να υλοποιηθεί με την χρήση της fgets με παράμετρο stdin καθώς θα διαβαζε ενα string δεδομένων. Όταν όμως προέκυψε η ανάγκη της δυναμικής κατανομής της μνήμης που θα χρησιμοποιηθεί για την αποθήκευση του string εισόδου, η fgets αντικαταστάθηκε απο την fgetc μέσα σε loop η οποία διαβάζει κάθε χαρακτήρα απο την είσοδο ξεχωριστά μέχρι να φτάσει σε newline ή EOF, και παράλληλα κάνει δυναμική κατανομή της μνήμης ανάλογα με τον αριθμό των χαρακτήρων που εχουν αποθηκευτεί στο string εισόδου με μεγαλύτερη ευκολία.

Μετά την επιτυχή προσπέλαση της εισόδου στο πρόγραμμα το μόνο που είχε απομείνει ηταν η κλήση της fork και επειτα η εκτέλεση της εντολής μέσω της συναρτησης execvp. Η υλοποίηση του προβλήματος αυτού δεν ήταν δύσκολη σε αυτό το σημείο καθώς η execvp δεν είχε παρα μόνο να εκτελέσει μία εντολη χωρίς καθολου flags, οπότε αρχικα το string εισόδου ηταν και η παράμετρος της execvp. Τοποθετώντας ενα while loop στην main με συνθήκη εξόδου το string “exit” στην είσοδο ολοκληρώθηκε το mysh1.

Mysh2

Για το mysh2 ο πρώτος στόχος ήταν η εκτέλεση των εντολών με υποστήρηξη παραμέτρων, καθώς ηταν επέκταση της λειτουργίας του mysh1 και βασιζόταν αρκετά στον υπάρχων κώδικα.

Για να γίνει αυτό χρειάστηκε μια πλήρη κατανόηση του πώς δουλεύει η execvp. Εφόσον χρειάζεται ώς παραμέτρους το ονομα της εντολης και εναν πίνακα string που περιλαμβάνει ολες τις παραμέτρους της εντολής, προστέθηκε στο shell η συνάρτηση get_args που με την χρήση της strtok σπάει το string εισόδου σε πολλα string όπου υπάρχει κενό ή χαρακτήρας στήλης, και τα κατανέμει όλα σε ένα char pointer array και τα επιστρέφει σε πίνακα argv. Έτσι η execvp δεν έχει παρα μόνο να λάβει ώς παραμέτρους τις argv[0] και argv για να λειτουργήσει.

Η υλοποίηση της εντολής cd εγινε όλη και αυτή μεσα σε μία συνάρτηση. Εκτός απο την κλήση της chdir η cd έπρεπε να δέχεται και relative paths. Μετά απο κάποιες δοκιμές αποδείχθηκε οτι τα . Και .. δεν χρειαζόταν να υλοποιηθουν στο shell καθως αναγνωρίζονται ως path στα ίδια τα linux. Για την ολοκλήρωση της cd χρειαζόταν μόνο η ικανότητα μεταφοράς σε subdirectory χωρίς full path και η εφαρμογή του home directory. Η πρώτη λύση και αυτή που δούλεψε ήταν ο έλεγχος του πρώτου χαρακτήρα του directory, εάν δεν ειναι / ή ~ τοτε γινεται προσκόληση του σε νέο string που μπροστά του έχει τους χαρακτήρες “./” έτσι ωστε να μεταφέρεται σε subdirectory επιτυχώς. Εαν ο πρώτος χαρακτήρας ειναι ~ τοτε το directory εισόδου γίνεται προσκόλληση στο home directory.

Το πρόβλημα που προέκυψε στο τέλος ητα το γεγονός οτι η cd δεν μπορούσε να λειτουργησει μεσω νέου process απο την fork σαν τις εντολες της execvp οπως εγινε αρχικα προσπάθεια, γιατι η chdir αλλάζει το directory του child process χωρις να αλλαζει αυτο του γονέα. Συνεπώς η cd

υλοποιείται μεσω του γονέα και ο ρόλος του έπαψε να είναι μονάχα να δημιουργει διεργασίες μεσω της fork, αλλα παραλληλα και να εκτελει τις εντολες του shell οπως cd και exit.

Mysh3

Εδώ παρ'όλο που η μόναδικη νέα λειτουργία του shell θα ήταν να υλοποιεί ενα pipe μονο, χρειάστηκε να γίνουν αρκετές αλλαγές στις προηγούμενες συναρτήσεις για να είναι έτοιμο το shell για αυτήν την λειτουργία.

Αρχικά θα έπρεπε να υπάρχει κάποιος τρόπος να διαχωριστεί η εντολή εισόδου σε δύο μέλη, ένα για κάθε end του pipe. Για αυτόν τον λόγο η argv επανονομάστηκε σε args και λειτουργέι ως είσοδος σε μια νέα συνάρτηση που ονομάζεται set_argv. Η συνάρτηση αυτή δέχεται ως είσοδο το pointer array που προκύπτει απο την get_args, και τις τοποθετεί σε εναν δισδιάστατο πίνακα δεικτών char argv με τετοιο τρόπο έτσι ώστε κάθε φορά που εμφανίζεται ο χαρακτήρας "|" οι εντολές να αποθηκέυονται στην επόμενη γραμμή της argv. Με αυτον τον τροπο η καθε γραμμη αντιστοιχεί σε ενα μέλος του pipe και περιέχει την εντολή προς εκτέλεση και τις παραμέτρους της.

Επιπλέον χρειάστηκε να γίνουν κάποιες αλλαγές και στον parser καθώς η άσκηση απαιτεί το shell να μπορει να υλοποιήσει pipes ακομα και αν το συμβολο τους "|" δεν διαχωριζεται απο κενά απο τις αλλες εντολές απαραίτητα. Έτσι η parser περα απο την κανονικη της λειτουργια προσθέτει και εναν κενό χαρακτήρα πρίν και μετά τον χαρακτήρα "|" έτσι ωστε όταν περάσει το string εισόδου στην get_args, αυτη θα χωρίσει σίγουρα το string εκει που υπαρχει pipe.

Μετά απο αυτά το shell ήταν έτοιμο να υλοποιήσει pipes, μέσω του child process που προκύπτει απο την κληση της fork στην main. Εκεί γίνεται έλεγχος για την ύπαρξη pipe και εάν υπάρχει, καλούνται δύο επιπλέον process απο την fork, ένα για κάθε μελος του pipe.