ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ

Ονοματεπώνυμο: Παναγιώτης Ντενέζος

Α.Μ.: 5853

- **Βασικό στήσιμο προγράμματος**

  Με την εκτέλεση του προγράμματος εμφανίζεται στην οθόνη ο χαρακτήρας "**$**" που υποδηλώνει ότι το shell είναι έτοιμο αν δεχτεί εντολή. Το διάβασμα της εντολής γίνεται με χρήση της συνάρτησης *getline().* Ο λόγος που επιλέχτηκε η συγκεκριμένη συνάρτηση είναι για να επιτυγχάνεται δυναμική δέσμευση της εισόδου. Με χρήση της συνάρτησης *malloc()* δεσμεύεται αρχικά ένας μικρός χώρος (συγκεκριμένα το μέγεθός ενός χαρακτήρα) και έπειτα με χρήση της συνάρτησης *realloc()* επαναδεσμεύεται τόσος χώρος όσο χρειάζεται για να αποθηκευτεί στην πρώτη γραμμή του πίνακα token η εντολή που δώσαμε. Πριν γίνει αυτό με χρήση της συνάρτησης *strtok()* η είσοδος χωρίζεται σε εντολές με συνθήκη διαχωρισμού το "\n" (και τον χαρακτήρα "|" σε περίπτωση pipes) και αποθηκεύεται μια εντολή σε κάθε γραμμή του πίνακα token. Στο τέλος κάθε εντολής αποθηκεύεται και η τιμή NULL για να δηλώσουμε το τέλος του string. Στη συνέχεια γίνεται έλεγχος για το πόσες εντολές δόθηκαν σαν είσοδος από τον χρήστη. Υπάρχουν 3 περιπτώσεις. Ο χρήστης να έδωσε:
  - Καμία εντολή. Η περίπτωση δηλαδή που απλά πατάει το κουμπί του enter σαν είσοδο.
  - Μια εντολή. Η απλή περίπτωση.
  - Περισσότερες από μια εντολές. Η περίπτωση των pipes.

- **Υλοποίηση της εκτέλεσης των εντολών**

  Σε αυτή την περίπτωση πάλι με χρήση της συνάρτησης *strtok()* γίνεται διαχωρισμός αυτή την φορά για κάθε γραμμή του πίνακα token (δηλαδή για κάθε εντολή ξεχωριστά) με συνθήκη διαχωρισμού τους χαρακτήρες " " (κενό) και "/t" (tab). Ουσιαστικά δημιουργείται ένας πίνακας τριών διαστάσεων εκ των οποίων οι δυο διαστάσεις περιέχουν τις γραμμές εντολών που δόθηκαν από τον χρήστη

και στη τρίτη διάσταση περιέχουν με δυναμικό τρόπο την εντολή διαχωρισμένη σε βασικό κομμάτι εντολής και παραμέτρους.
Υπάρχουν όπως αναφέρθηκε και παραπάνω 2 περιπτώσεις.

1. Η είσοδος να είναι μια εντολή

   Σε αυτή την περίπτωση απλά με χρήση της συνάρτησης *fork()* δημιουργούμε ένα παιδί και με χρήση της συνάρτησης *execvp()* εκτελούμε την εντολή αν και εφόσον υπάρχει στο περιβάλλον της εντολής περιβάλλοντος PATH.

2. Η είσοδος να είναι παραπάνω από μια εντολές (pipes)

   Σε αυτή την περίπτωση, αρχικά δημιουργείται ένας πίνακας ακεραίων όσος και ο αριθμός των pipes. Στη συνέχεια με την εντολή συστήματος *pipe()*, μετατρέπεται σε κανάλι για διαδικαστική επικοινωνία. Με χρήση της εντολής for δημιουργούνται τόσα παιδιά όσα και οι εντολές με τον τρόπο που εξηγήθηκε παραπάνω για το ένα παιδί. Η λογική που υλοποιείται είναι ότι στη γενική περίπτωση το i-οστό παιδί δέχεται την είσοδο του την θέση (2i - 2) του πίνακα και γράφει την έξοδο του στη θέση (2i + 1) του ιδίου πίνακα . Η πρώτη εντολή παίρνει είσοδο από το stdin και γράφει την έξοδο του στην θέση (2i + 1) , ενώ η τελευταία παίρνει είσοδο από τη θέση (2i - 2) και γράφει την έξοδο του στο stdout. Για αυτό και δημιουργήθηκες ξεχωριστός έλεγχος για αυτές τις 2 θέσεις. Τέλος με επαναληπτική διαδικασία ο πατέρας κλείνει όλα τα pipes.

Και στις 2 παραπάνω περιπτώσεις το shell περιμένει τον τερματισμό του ενός ή των πολλών παιδιών μέχρι να ξαναεμφανίσει τον χαρακτήρα “$” στην οθόνη και να είναι έτοιμο αν δεχτεί την επόμενη εντολή.

- **Υλοποίηση εντολών cd & exit**

  Στην περίπτωση που ο χρήστης δίνει σαν είσοδο μια μόνο εντολή γίνεται έλεγχος για το αν αυτή η εντολή είναι η εντολή “exit”. Αν ο χρήστης δώσει την εντολή “exit” τότε εκτελείται η εντολή της C exit(0) και το πρόγραμμα τερματίζεται με επιτυχία.

Στην περίπτωση που δοθεί σαν είσοδος η εντολή “cd”, τότε το πρόγραμμα εντοπίζει το path στο οποίο βρισκόμαστε τώρα και προσθέτει στο τέλος αυτό που δόθηκε σαν όρισμα στην “cd”. Αν δεν υπάρχει αυτός ο φάκελο τότε χρησιμοποίει το path που δόθηκε και το ψάχνει από την αρχή.

- **Προβλήματα που υπήρξαν**

  Το πρόβλημα που εμφανίσθηκε ήταν η δυναμική διαχείριση του χώρου, αλλά με μια γρήγορη επανάληψη στην C στο κομμάτι των pointers το πρόβλημα αυτό λύθηκε εύκολα. Τα προβλήματα που υπήρξαν αφορούσαν επί το πλείστων την δημιουργία των pipes. Η υλοποίηση που έγινε κατά την γνώμη μου είναι η πιο εύκολα κατανοητή και ήταν το αποτέλεσμα μετά από πολλές ώρες ψάξιμο και κατανόησης διαφορετικών τρόπων υλοποίησης των pipes.