ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ

ΑΝΑΦΟΡΑ 1ης ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

ΑΝΑΣΤΑΣΙΟΣ ΜΠΕΝΟΣ 3130141 – ΕΥΘΥΜΙΟΣ ΤΣΟΠΕΛΑΣ 3130210

Περιγραφή Προγράμματος:

Για να υλοποιήσουμε τα ζητούμενα χρησιμοποιήσαμε μια μέθοδο (ShellyLoop()) προκειμένου να προσομοιώσουμε τον τρόπο που λειτουργει το Unix Shell . Πιο συγκεκριμένα , το πρόγραμμα μας δέχεται την τρέχουσα εντολή απο το terminal (read_input()), την επεξεργάζεται με προκαθορισμένα κριτήρια (cook()) και την προωθεί (orient()) στον κατάλληλο μηχανισμό προς εκτέλεση. Μετα την εκτέλεση της εντολής , το πρόγραμμα , αναμένει νέα είσοδο απο τον χρήστη.

ΣΗΜΑΝΤΙΚΗ ΣΗΜΕΙΩΣΗ :

Γνωρίζουμε ότι η υλοποίηση μας δεν συναδει απολυτα με τον τρόπο που η εκφώνηση απαιτεί, τον διαχωρισμό των φλοιών σε πέντε αρχεία.
Πιο συγκεκριμενα , έχουμε γενικότερες μεθόδους που καλύπτουν τα ζεύγη ερωτήματων (1)-2-3 και 4-5 . Ασφαλώς το πρόγραμμα αντιλαμβάνεται ποια είναι η τρέχουσα περίπτωση φλοιού (και τη δηλώνει με τη κατάλληλη προτροπη auebsh*>). Στην εκφώνηση αναφέρεται ότι απαιτούνται πέντε αρχεία (ένα για κάθε φλοίο). Στη δική μας περίπτωση ,επιλέξαμε να αποφύγουμε την επαναχρησιμοποιηση κώδικα και να βασιστούμε καθολοκληρίαν στις τρεις μεθόδους που καλύπτουν άρτια και μοναδικά τις πέντε ζητούμενες περιπτώσεις.

Παραθέτουμε , κάτωθεν , την αναλυτική περιγραφή της λειτουργίας κάθε συνάρτησης ανά αρχείο:

p3130141-p3130210-auebsh-common.c :

Η μέθοδος **ShellyLoop** αποτελείται από μια βασική δομή επανάληψης (while) που τρέχει μέχρις ότου τερματιστεί ο φλοιός . Εντός της while πραγματοποιούνται κλήσεις τριών διαφορετικών συναρτήσεων που ακολουθούν.

Η μέθοδος **read_input** αποθηκεύει κάθε χαρακτήρα της εισόδου του χρήστη σε έναν πίνακα, τύπου char, μέχρις ότου βρεί τον χαρακτήρα αλλαγής γραμμής. Επίσης είναι ικανή να τερματίσει το πρόγραμμα , αν δεχθεί την μακροεντολή Ctrl-D. Με την ολοκλήρωση της συγκεκριμένης μεθόδου επιστρέφεται ο προαναφερθέν πίνακας.

Η μέθοδος **cook** δέχεται σαν όρισμα τον πίνακα της read_input και διαχωρίζει τις λέξεις που απαρτίζουν την εντολή. Αυτος ο διαμερισμός πραγματοποιείται κάθε φορά που συναντά τον χαρακτήρα του κενού. Αποθηκεύει κάθε ξεχωριστή λέξη σε έναν πίνακα tokarray (char **) , τον οποίο και επιστρέφει.

Η μέθοδος **orient** είναι υπεύθυνη για δύο λειτουργίες: Αναγνωρίζει αν υπάρχουν σωληνώσεις και ανακατευθύνσεις στην τρέχουσα εντολή με σκοπό να εκτυπώσει τη κατάλληλη προτροπή (auebsh*>) για τον φλοιό που ταιριάζει. Απο την άλλη πραγματοποιεί κλήση των “μεθόδων εκτέλεσης” (pre_pip() ,exc_worker()) με βάση τα άνωθεν κριτήρια.

p3130141-p3130210-auebsh-1-2-3.c :

Η μέθοδος **exc_worker()** είναι υπεύθυνη για την επίλυση των τριών πρώτων ερωτημάτων της εργασίας. Με τη χρήση fork εκτελεί απλές εντολές καθώς και εντολές που εμπεριέχουν ανακατευθύνσεις. Πιο συγκεκριμένα , σε περίπτωση που υπάρχουν ανακατευθύνσεις αφαιρούνται οι ειδικοι χαρακτήρες ανακατεύθυνσης (<,>). Εν συνεχεία , με τη χρήση της dup2 αναδρομολογούμε κατάλληλα τους file descriptors ωστε να διαβάζουμε απο αρχείο και αντίστοιχα να γράφουμε. Αυτό επιτυγχάνεται με την αποθήκευση των στοιχείων (ονόματα αρχείων) πριν το < και μετά το > σε μεταβλητές in ,out αντιστοιχα. Ο πίνακας που

θα προκυψει και εκτελεστεί από την excvp εμπεριέχει μονάχα την βασικη εντολή. Πχ. Αν δοθεί στο πρόγραμμα ως είσοδος η εντολη sort –u < input.txt > output.txt , βάσει της λογικής που περιγράψαμε άνωθεν η execvp θα εκτελεσει απλώς την εντολη sort –u έχοντας εν τουτοις ενημερωμένους τους file descriptors στα απαιτούμενα αρχεία. Με αυτό τον τρόπο , καθίσταται ικανή η εκτέλεση της εντολής μέ ή χωρίς παράμετρο.

p3130141-p3130210-auebsh-4-5.c :

Η μέθοδος **pre_pip** καλείται σε περίπτωση που η orient έχει εντοπίσει έστω και μία σωλήνωση (|) και δέχεται ως όρισμα τον πίνακα που έχει παράξει η cook() δημιουργώντας έναν εκλεπτισμένο πίνακα (char ***) ο οποίος εμπεριέχει όλες τις εντολες προς εκτέλεση. Ασφαλώς, η δημιουργία αυτού του ιδιαίτερου πίνακα απαιτούσε τον εντοπισμό των σωληνώσεων και με τη βοήθεια κάποιων pointers επιτύχαμε την αντιγραφη των “προτάσεων” σε αυτόν για τη καλύτερη και ταυτόχρονη διαχείριση τους. Από τη στιγμή που ολοκληρωθεί η δημιουργία αυτού του πίνακα είμαστε έτοιμοι να καλέσουμε τη μέθοδο piping().

Η μέθοδος **piping** δέχεται σαν είσοδο τον πίνακα που δημιουργεί η pre_pip() και είναι υπεύθυνη για την επίλυση των δυο τελευταίων ζητημάτων της εργασίας. Για άλλη μια φορά χρησιμοποιήσαμε τη fork και με τη βοήθεια pipes επιτύχαμε επικοινωνία μεταξύ των διεργασιών. Πιο συγκεκριμένα όταν εκτελείται ο κώδικας του παιδιού κλείνουμε την πλευρά του pipe που είναι υπεύθυνη για το read .Ενώ όταν εκτελείται ο κώδικας του πατέρα κλείνουμε την άλλη πλευρά. Εν παραλλήλω , με τη χρήση της dup2 αναδρομολογούμε κατάλληλα τους file descriptors ώστε το αποτέλεσμα της εκτέλεσης της εκάστοτε εντολής(**πρότασης**) να αποτελεί input για την επόμενη. Ασφαλώς , η εκτέλεση της κάθε πρότασης αναλαμβάνεται απο την exc_worker() που περιγράψαμε λεπτομερώς παραπάνω ωστε να καλύπτουμε περιπτώσεις που μέρος των σωληνώσεων εμπεριέχουν redirections.