Οδηγίες και Συμβουλές για την Ομαλή Διεκπεραίωση της Εργασίας

Η εργασία θα πρέπει να πραγματοποιηθεί σε βήματα. Ο κάθε φοιτητής είναι υπεύθυνος να αποφασίσει ποια βήματα ταιριάζουν στον τρόπο εργασίας του. Στη συνέχεια, παρατίθεται μια δυνητική ακολουθία βημάτων που θα μπορούσε να ακολουθηθεί για την ομαλή διεκπεραίωση της εργασίας. Σε κάθε βήμα θα πρέπει να κρατάτε αντίγραφα του κώδικα που έχετε φτιάξει και να προχωράτε στην παραγωγή του κώδικα του επόμενου βήματος σε ένα νέο αρχείο, ώστε να μπορείτε να χρησιμοποιήσετε τον κώδικα προηγούμενων βημάτων, αν χρειαστεί.

Βήμα 1: Ξεκινήστε υλοποιώντας ένα μέρος του γεγονότος R (Register) και ένα μέρος του γεγονότος D. Για την υλοποίηση του γεγονότος αυτού πρέπει να υλοποιήσετε τη λίστα λογαριασμών. Η λίστα αυτή είναι απλά συνδεδεμένη και μη ταξινομημένη και έχει κόμβο φρουρό. Ξεκινήστε αγνοώντας το γεγονός πως η λίστα λογαριασμών έχει κόμβο φρουρό και γράψτε κώδικα (σε ένα δικό σας αρχείο) ο οποίος θα υλοποιεί μια απλά συνδεδεμένη μη-ταξινομημένη λίστα.

Φτιάξτε διαδικασίες ALInsert, ALDelete και ALLookUp για μια μη-ταξινομημένη απλά-συνδεδεμένη λίστα, προκειμένου να πραγματοποιείτε εισαγωγές, διαγραφές και αναζητήσεις, αντίστοιχα, στη λίστα αυτή. Τα στοιχεία της λίστα αυτής μπορούν να είναι τύπου account_st (όπου στη φάση αυτή, το πεδίο myphotos θα είναι ίσο με NULL). Φροντίστε ώστε η ALLookUp() να επιστρέφει δείκτη στον κόμβο που αναζητείται (αν βρεθεί) ή NULL (αν δεν βρεθεί). Αυτό θα σας βοηθήσει αργότερα, όταν θα πρέπει να φτιάξετε τη λίστα οπαδών κάθε στοιχείου της λίστας φωτογραφιών καθώς και τον πίνακα TAGS αυτού του στοιχείου.

Δημιουργήστε μια διαδικασία ALPrint() για την εκτύπωση των στοιχείων της λίστας η οποία θα σας βοηθήσει να ελέγξετε την ορθότητα των διαδικασιών που περιγράφονται παραπάνω. Δημιουργήστε τον παραπάνω κώδικα σε ένα δικό σας αρχείο και φτιάξτε μια δική σας main() για να ελέγξετε ότι ο κώδικας που υλοποιεί τις λειτουργίες μιας απλά συνδεδεμένης ταξινομημένης λίστας λειτουργεί σωστά. Όταν είστε σίγουροι για αυτό συνεχίστε στο επόμενο βήμα.

Βήμα 2: Αντιγράψτε το αρχείο που δημιουργήσατε στο βήμα 1 σε ένα νέο αρχείο και κάντε τροποποιήσεις σε αυτό ώστε να υλοποιήσετε τον κόμβο φρουρό. Η λίστα θα περιέχει αρχικά τον κόμβο φρουρό και θα υπάρχει ένας ακόμη δείκτης που θα δείχνει σε αυτόν. Ο κόμβος φρουρός θα είναι πάντα ο τελευταίος κόμβος στη λίστα. Η LookUp() θα πρέπει να τροποποιηθεί ώστε να είναι πιο αποτελεσματική, βάσει όσων έχετε διδαχθεί για τον κόμβο φρουρό. Η Insert() και η Delete() ίσως να χρειάζεται να τροποποιηθούν επίσης.

Απομονώστε στα test_files που σας παρείχαν οι βοηθοί τα γεγονότα τύπου R και εκτελέστε το πρόγραμμά σας μόνο για τέτοια γεγονότα. Το γεγονός τύπου R έχει βέβαια μόνο μερικά υλοποιηθεί αφού δεν υπάρχουν ακόμη οι λίστες προσωπικών φωτογραφιών του κάθε λογαριασμού. Έτσι, στο βήμα αυτό απλά ελέγχετε αν τα στοιχεία της λίστας λογαριασμών ενημερώνονται σωστά.

Βήμα 3: Συνεχίστε με την υλοποίηση ενός μέρους των γεγονότων U και B, υλοποιώντας τη λίστα φωτογραφιών. Η λίστα αυτή είναι διπλά-συνδεδεμένη, ταξινομημένη (ως προς το πεδίο pid) και κάθε στοιχείο της είναι τύπου photo_st.

Γράψτε κώδικα που θα υλοποιεί τη λίστα αυτή σε ένα εντελώς νέο αρχείο. Αρχικά, υλοποιήστε μια απλούστερη έκδοση της λίστας αυτής στην οποία ο δείκτης likes κάθε struct photo_st είναι NULL και το ίδιο και ο πίνακας TAGS του struct αυτού.

Φτιάξτε διαδικασίες SDLInsert, SDLDelete και SDLLookUp για μια ταξινομημένη διπλά συνδεδεμένη λίστα, προκειμένου να πραγματοποιείτε εισαγωγές, διαγραφές και αναζητήσεις, αντίστοιχα, στη λίστα αυτή. Δημιουργήστε μια διαδικασία SDLPrint() για την εκτύπωση των στοιχείων της λίστας η οποία θα σας βοηθήσει να ελέγξετε την ορθότητα των διαδικασιών που περιγράφονται παραπάνω. Τροποποιήστε κατάλληλα τη main() του Βήματος 1 για να ελέγξετε ότι ο κώδικας που υλοποιεί τις λειτουργίες μιας διπλά συνδεδεμένης, ταξινομημένης λίστας λειτουργεί σωστά. Όταν είστε σίγουροι για αυτό συνεχίστε στο επόμενο βήμα.

Απομονώστε στα test_files που σας παρείχαν οι βοηθοί τα γεγονότα τύπου U και εκτελέστε το πρόγραμμά σας μόνο για τέτοια γεγονότα. Το γεγονός τύπου U έχει βέβαια μόνο μερικά υλοποιηθεί αφού δεν υπάρχουν ακόμη οι λίστες προσωπικών φωτογραφιών του κάθε λογαριασμού, κλπ. Έτσι, στο βήμα αυτό απλά ελέγχετε αν τα στοιχεία της λίστας φωτογραφιών ενημερώνονται σωστά.

Βήμα 4: Υλοποιήστε το γεγονός τύπου T, δηλαδή αποκαταστήστε την ορθή λειτουργία του πίνακα TAGS σε κάθε κόμβο της λίστας φωτογραφιών. Για κάθε τέτοιο γεγονός απαιτείται (1) να καλείται την ALLookUp() και να αποθηκεύετε το δείκτη που αυτή επιστρέφει σε μια βοηθητική μεταβλητή, (2) να καλείται την SDLLookUp() για να βρίσκετε το στοιχείο της λίστας φωτογραφιών στο οποίο αντιστοιχεί το αναγνωριστικό <pid> και να προσθέτετε στο επόμενο ελεύθερο στοιχείο του πίνακα TAGS της φωτογραφίας αυτής, τον δείκτη που επέστρεψε η ALLookUp().

Βήμα 5: Υλοποιήστε ένα ακόμη μέρος του γεγονότος D, δημιουργώντας μια ρουτίνα, έστω DeleteReferencesToAccount(), η οποία θα διατρέχει τη λίστα φωτογραφιών και για κάθε στοιχείο της, αν κάποιο στοιχείο του πίνακα TAGS του στοιχείου αυτού δείχνει στο στοιχείο με αναγνωτιστικό <aid> της λίστας λογαριασμών, θα το αντικαθιστά με NULL.

Βήμα 6: Υλοποιήστε τη λίστα likes κάθε στοιχείου της λίστας φωτογραφιών. Αυτή είναι μια μη-ταξινομημένη απλά-συνδεδεμένη λίστα και άρα μια λίστα ίδιου τύπου με αυτή που δημιουργήσατε στο Βήμα 1. Ξεκινήστε αντιγράφοντας τον κώδικα που γράψατε στο Βήμα 1 και κάνοντας κατάλληλες τροποποιήσεις ώστε η λίστα τώρα να περιέχει στοιχεία τύπου likes_st. Στη φάση αυτή, το πεδίο aptr του struct αυτού θα μπορούσε να είναι ακέραιος και να αποθηκεύει το αναγνωριστικό του λογαριασμού αντί για δείκτη στο κατάλληλο στοιχείο της λίστας λογαριασμών.

Στη συνέχεια, συνενώστε τον κώδικα αυτό με τον κώδικα που δημιοουργήσατε στο Βήμα 5 για να μετατρέψετε τη λίστα φωτογραφιών σε μια λίστα από λίστες.

Πειραματιστείτε με το γεγονός τύπου L. Στην παρούσα φάση, κάθε τέτοιο γεγονός θα πρέπει να προκαλεί τις εξής ενέργειες: (1) αναζήτηση του στοιχείου με αναγνωριστικό <pid> στη λίστα φωτογραφιών, (2) εισαγωγή ενός νέου στοιχείου στη λίστα οπαδών της φωτογραφίας αυτής. Προσέξτε πως το πρώτο στοιχείο της λίστας αυτής δεικτοδοτείται από το πεδίο likes του struct της φωτογραφίας.

Βήμα 7: Τροποποιήστε το πεδίο aptr του struct likes_st να είναι δείκτης και όχι ακέραιος. Αυτό προϋποθέτει πως θα καλείται επιπρόσθετα και η ALLookUp() για να γίνεται αναζήτηση του λογαριασμού με αναγνωριστικό <aid> στη λίστα λογαριασμών.

Βήμα 8: Ολοκληρώστε το γεγονός E συνδυάζοντας τις Print() που δημιουργήσατε για την αποσφαλμάτωση της λίστας φωτογραφιών και της λίστας οπαδών.

Βήμα 9: Προχωρήστε με την υλοποίηση του γεγονότος D έτσι ώστε τώρα η διαγραφή ενός λογαριασμού να οδηγεί στη διαγραφή και όλων των αναφορών στο λογαριασμό αυτό οι οποίες βρίσκονται στις λίστες οπαδών των διαφόρων φωτογραφιών. Για να το κάνετε αυτό, ο απλούστερος τρόπος είναι να κάνετε μικρές τροποποιήσεις στην DeleteReferencesToAccounts() ώστε αυτή να καλεί για κάθε στοιχείο της λίστας φωτογραφιών που διασχίζει και μια ρουτίνα διαγραφής στη λίστα οπαδών που αντιστοιχεί σε αυτή τη φωτογραφία. Η ρουτίνα αυτή αναζητά στοιχείο με ανανωριστικό <aid> στη λίστα αυτή (κάνοντας de-reference) και αν βρει κάποιο τέτοιο στοιχείο το διαγράφει.

Βήμα 10: Υλοποιήστε τη λίστα προσωπικών φωτογραφιών ενός λογαριασμού. Η λίστα αυτή είναι ταξινομημένη (ως προς το πεδίο pid) και απλά-συνδεδεμένη.

Γράψτε κώδικα που θα υλοποιεί τη λίστα αυτή σε ένα εντελώς νέο αρχείο. Φτιάξτε διαδικασίες SSLInsert, SSLDelete και SSLLookUp για μια ταξινομημένη απλά-συνδεδεμένη λίστα, προκειμένου να πραγματοποιείτε εισαγωγές, διαγραφές και αναζητήσεις, αντίστοιχα, στη λίστα αυτή. Δημιουργήστε μια διαδικασία SSLPrint() για την εκτύπωση των στοιχείων της λίστας η οποία θα σας βοηθήσει να ελέγξετε την ορθότητα των διαδικασιών που περιγράφονται παραπάνω. Τροποποιήστε κατάλληλα τη main() του Βήματος 1 για να ελέγξετε ότι ο κώδικας που υλοποιεί τις λειτουργίες μιας απλά-συνδεδεμένης, ταξινομημένης λίστας λειτουργεί σωστά. Όταν είστε σίγουροι για αυτό συνεχίστε στο επόμενο βήμα.

Βήμα 11: Συνενώστε τους κώδικες που δημιουργήσατε στα βήματα 2 και 7 για να τροποποιήσετε τη λίστα λογαριασμών ώστε τώρα αυτή να γίνει λίστα από λίστες. Ολοκληρώστε την υλοποίηση του γεγονότος U, προσθέτοντας την φωτογραφία με αναγνωριστικό <pid> στη λίστα προσωπικών φωτογραφιών του λογαριασμού με αναγνωριστικό <aid>.

Υλοποιήστε τα γεγονότα τύπου P και A.

Βήμα 12: Ολοκληρώστε την υλοποίηση των γεγονότων D και B.

Βήμα 13: Υλοποιήστε το γεγονός τύπου F. Βεβαιωθείτε πως το γεγονός εκτελείται σωστά.

Βήμα 14: Υλοποιήστε το γεγονός τύπου M. Προσέξτε ώστε η χρονική πολυπλοκότητα του αλγορίθμου συνένωσης (merge) που υλοποιήσατε να είναι η ζητούμενη.

Βήμα 15: Υλοποιήστε το γεγονός τύπου S. Προσέξτε ώστε η χρονική πολυπλοκότητα του αλγορίθμου διαχωρισμού (split) που υλοποιήσατε να είναι η ζητούμενη.

Βήμα 16: Ελέγξετε την ορθότητα του κώδικα που δημιουργήσατε εκτελώντας τον κώδικά σας σε όλα τα αρχεία εκτέλεσης που θα σας παρέχουν οι βοηθοί του μαθήματος. Επιπρόσθετα, δημιουργήστε τα δικά σας αρχεία γεγονότων για να ελέγξετε με περισσότερη ακρίβεια την ορθότητα του κώδικά σας.