HY240: Δομές Δεδομένων
Χειμερινό Εξάμηνο – Ακαδημαϊκό Έτος 2008-09
Παναγιώτα Φατούρου

Προγραμματιστική Εργασία
3ο Μέρος

Ημερομηνία Παράδοσης: ~~Τρίτη, 6 Ιανουαρίου 2009, ώρα 23:59~~ **(Νέα ημερομηνία: Τρίτη, 13/1, ώρα 23:59).**
Τρόπος Παράδοσης: Χρησιμοποιώντας το πρόγραμμα submit.

Στο 3ο μέρος της προγραμματιστικής εργασίας ζητείται να γίνουν τροποποιήσεις στο πρόγραμμα που προέκυψε κατά την υλοποίηση του 2ου προγραμματιστικού καθήκοντος του 2ου μέρους. Το πρόγραμμα που θα προκύψει θα περιγράφει (προσομοιώνει) και πάλι τη λειτουργία ενός διομότιμου συστήματος (το οποίο ακολουθεί τους κανόνες λειτουργίας που έχουν περιγραφεί στο 1ο μέρος). Το 3ο μέρος της εργασίας αποτελείται από τρία προγραμματιστικά καθήκοντα.

1ο Προγραμματιστικό καθήκον: Υλοποίηση Πινάκων Κατακερματισμού (Hash Tables)

Οι πληροφορίες για τους κόμβους που συμμετέχουν κάθε χρονική στιγμή στο σύστημα αποθηκεύονται, στο *δένδρο κόμβων* (NodeTree) (δηλαδή σε ένα ταξινομημένο διπλά-συνδεδεμένο δένδρο), όπως είχε περιγραφεί στο 2ο μέρος της εργασίας. Για κάθε κόμβο n διατηρούνται οι πληροφορίες που περιγράφηκαν στο 2ο μέρος της εργασίας. Στο μέρος αυτό ωστόσο τα αρχεία που διατηρεί κάθε κόμβος θα αποθηκεύονται σε έναν πίνακα κατακερματισμού. Ζητείται να υλοποιήσετε τις εξής δύο τεχνικές κατακερματισμού:

1. **Μέθοδος των ταξινομημένων ξεχωριστών αλυσίδων.** Η μορφή των δομών δεδομένων που χρησιμοποιεί το πρόγραμμα στην περίπτωση αυτή φαίνεται στο Σχήμα 1.
2. **Μέθοδος του ταξινομημένου διπλού κατακερματισμού.** Η μορφή των δομών δεδομένων που χρησιμοποιεί το πρόγραμμα στην περίπτωση αυτή φαίνεται στο Σχήμα 2.

Οι συναρτήσεις κατακερματισμού (πρωτεύουσα και δευτερεύουσα) που θα χρησιμοποιήσετε θα πρέπει να επιλέγονται τυχαία από μια καθολική οικογένεια συναρτήσεων κατακερματισμού (δηλαδή ζητείται να υλοποιήσετε μια καθολική οικογένεια συναρτήσεων κατακερματισμού, βάσει της θεωρίας που έχετε διδαχθεί στο μάθημα).

Επιπρόσθετα, για κάθε αρχείο κρατείται τώρα και το μέγεθος του (πεδίο size των κόμβων του πίνακα κατακερματισμού αρχείων).

Το πρόγραμμα θα πρέπει να εκτελείται καλώντας την εντολή

run <m> <p> <input-file> <number_of_events>

όπου run είναι το όνομα του εκτελέσιμου αρχείου του προγράμματος, <m> είναι ένας ακέραιος που καθορίζει το μέγιστο πλήθος κόμβων στο σύστημα (το οποίο θα είναι 2^m), <p> είναι ο κοντινότερος πρώτος αριθμός στον 2^m που είναι μικρότερος του 2^m, <input-file> είναι το όνομα ενός αρχείου εισόδου που περιέχει γεγονότα των μορφών που περιγράφηκαν στο 1ο μέρος της εργασίας αλλά με χρονοσφραγίδες όπως περιγράφηκε στο 2ο μέρος της εργασίας (2ο προγραμματιστικό καθήκον) και <number_of_events> είναι το μέγιστο πλήθος γεγονότων που θα εκτελεστούν.

Η συμπεριφορά κάθε γεγονότος πρέπει να είναι αυτή που περιγράφεται στην εκφώνηση του $1^{ου}$ μέρους της εργασίας, ενώ οι χρονοσφραγίδες λειτουργούν με τον ίδιο ακριβώς τρόπο όπως στο $2^{ο}$ μέρος της εργασίας (δηλαδή θα πρέπει να αλλάξετε το αρχείο που προέκυψε κατά την υλοποίηση του $2^{ου}$ προγραμματιστικού καθήκοντος του $2^{ου}$ μέρους της εργασίας).Το πρόγραμμα θα πρέπει να τυπώνει αντίστοιχες πληροφορίες κατά την εκτέλεση κάποιου γεγονότος με αυτές που τυπώνονταν στο $2^{ο}$ μέρος της εργασίας.

Να ορίσετε με πρωτοβουλία σας τις απαραίτητες λεπτομέρειες, όπως το μέγεθος του πίνακα κατακερματισμού. Θεωρήστε ότι τα αναγνωριστικά των κλειδιών των αρχείων θα επιλεχθούν από το σύνολο {0,..,p-1}, όπου p είναι ο κοντινότερος πρώτος αριθμός στον αριθμό 2^m που είναι μικρότερος του 2^m.

$2^{ο}$ Προγραμματιστικό καθήκον: Ταξινόμηση

Προσθέστε ένα νέο γεγονός όπως περιγράφεται στη συνέχεια:

- S <id>: Γεγονός τύπου Sort το οποίο σηματοδοτεί την εκτύπωση των αρχείων που αποθηκεύονται στον κόμβο με αναγνωριστικό <id> **ταξινομημένα ως προς το size τους** (δηλαδή η εκτύπωση τυπώνει τα αρχεία του κόμβου <id> ταξινομημένα ως προς το μέγεθος τους). Η υλοποίηση αυτού του γεγονότος θα πρέπει να πραγματοποιηθεί ως εξής. Θα πρέπει πρώτα να δημιουργηθεί ένας πίνακας ο οποίος έχει τόσα στοιχεία όσα και τα αρχεία που είναι αποθηκευμένα στον κόμβο. Κάθε στοιχείο του πίνακα είναι ένα struct με δύο πεδία: id και size. Στη συνέχεια, ο πίνακας θα πρέπει να ταξινομείται ως προς το πεδίο size και τέλος να πραγματοποιείται η εκτύπωση (όπως περιγράφεται παρακάτω). Ως αλγόριθμο ταξινόμησης θα πρέπει να υλοποιήσετε είτε τον αλγόριθμο HeapSort() ή τον MergeSort().

  Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

  S <id> DONE
  DOCLIST = <fd_1, $size_1$>, <fd_2, $size_2$>, …, <fd_r, $size_r$>,

  όπου r είναι το πλήθος των αρχείων που είναι αποθκευμένα στον κόμβο με αναγνωριστικό <id> και για κάθε j, $1 \leq j \leq r$, fd_j είναι το αναγνωριστικό ενός αρχείου του πίνακα κατακερματισμού αρχείων του κόμβου και $size_j$ είναι το μέγεθος του αρχείου αυτού. Σημειώνεται τα αρχεία εμφανίζονται στη γραμμή «DOCLIST=…» σε φθίνουσα διάταξη ως προς το μέγεθος τους.

Το πρόγραμμα εκτελείται καλώντας την ίδια εντολή όπως περιγράφηκε παραπάνω για το $1^{ο}$ προγραμματιστικό καθήκον και τυπώνει ακριβώς τις ίδιες πληροφορίες μετά την εκτέλεση κάθε γεγονότος.

$3^{ο}$ Προγραμματιστικό καθήκον: Πειραματική Μελέτη

Στα διάφορα μέρη της προγραμματιστικής εργασίας έχετε υλοποιήσει ένα σύνολο από δομές δεδομένων. Πιο συγκεκριμένα, πληροφορίες για τους κόμβους αποθηκεύονται είτε σε λίστα ή σε ταξινομημένο δένδρο, ενώ τα αρχεία κάθε κόμβου αποθηκεύονται είτε σε λίστα ή σε πίνακα κατακερματισμού. Σε αυτό το προγραμματιστικό καθήκον ζητείται να μετρήσετε πειραματικά την απόδοση των σχεδιαστικών αυτών επιλογών. Ειδικότερα, θα πρέπει να συγκρίνετε τα εξής:

1. **Επιλογή 1**: Οι κόμβοι αποθηκεύονται σε λίστα και τα αρχεία κάθε κόμβου διατηρούνται επίσης σε λίστα (όπως στο $1^{ο}$ μέρος).
2. **Επιλογή 2**: Οι κόμβοι αποθηκεύονται σε ταξινομημένο δένδρο ενώ τα αρχεία κάθε κόμβου αποθηκεύονται σε λίστα (όπως στο $2^{ο}$ μέρος).

3. **Επιλογή 3**: Οι κόμβοι αποθηκεύονται σε ταξινομημένο δένδρο ενώ τα αρχεία κάθε κόμβου αποθηκεύονται σε πίνακα κατακερματισμού που ακολουθεί τη μέθοδο των ξεχωριστών ταξινομημένων αλυσίδων (όπως περιγράφεται στο τρέχον μέρος της εργασίας).
4. **Επιλογή 4**: Οι κόμβοι αποθηκεύονται σε ταξινομημένο δένδρο ενώ τα αρχεία κάθε κόμβου αποθηκεύονται σε πίνακα κατακερματισμού που ακολουθεί τη μέθοδο του ταξινομημένου διπλού κατακερματισμού (όπως περιγράφεται στο τρέχον μέρος της εργασίας).

Θα πρέπει να τροποποιήσετε κατάλληλα την διαδικασία main() του προγράμματός σας, ώστε αυτή να διαβάζει από το πληκτρολόγιο ποιες επιλογές θέλει να συγκρίνει ο χρήστης. Ανάλογα με τις επιλογές του χρήστη, θα πρέπει να καλείται ο κατάλληλος κώδικας και να προσμετρούνται διάφορα στατιστικά που περιγράφονται στη συνέχεια.

Για κάθε μια από τις επιλογές 1-4 που περιγράφηκαν παραπάνω θα πρέπει να μετράται το μέσο πλήθος προσπελάσεων που πραγματοποιούνται στις ακόλουθες περιπτώσεις:

- Εύρεση κόμβου.
- Εύρεση αρχείου στη δομή δεδομένων αρχείων ενός κόμβου (εδώ δεν προσμετρούνται οι προσπελάσεις μνήμης που απαιτούνται για την εύρεση του κατάλληλου κόμβου, αλλά μόνο εκείνες που πραγματοποιούνται για την εύρεση του αρχείου δεδομένου ότι ο κόμβος στον οποίο ανήκει η δομή δεδομένων αρχείων είναι γνωστός).
- Εύρεση αρχείου όπου προσμετρούνται οι συνολικές προσπελάσεις που πραγματοποιούνται κατά την αναζήτηση (τόσο στη δομή των κόμβων για να εντοπιστεί ο κατάλληλος κόμβος όσο και στη δομή των αρχείων του κόμβου αυτού).

Είναι αξιοσημείωτο ότι αναζητήσεις κόμβων ή αρχείων μπορεί να προκύπτουν κατά την εκτέλεση πολλών γεγονότων και θα πρέπει όλες αυτές οι αναζητήσεις να λαμβάνονται υπ' όψιν κατά τον υπολογισμό των στατιστικών (π.χ., οι περισσότεροι τύποι γεγονότων εμπεριέχουν τη λειτουργία της εύρεσης κόμβου και οι προσπελάσεις προκειμένου να πραγματοποιηθεί η εύρεση για κάθε τέτοιο γεγονός θα πρέπει να προσμετρούνται).

Στην περίπτωση που οι κόμβοι αποθηκεύονται σε ταξινομημένο δένδρο θα πρέπει επίσης να τυπώνεται το μέγιστο ύψος του δένδρου κατά τη διάρκεια της εκτέλεσης.

Επιπρόσθετα, εάν χρησιμοποιούνται πίνακες κατακερματισμού που ακολουθούν τη μέθοδο των ξεχωριστών ταξινομημένων αλυσίδων θα πρέπει να τυπώνονται τα εξής:

- Το μέσο μήκος των αλυσίδων (άθροισμα μηκών όλων των αλυσίδων όλων των πινάκων κατακερματισμού δια το συνολικό πλήθος αλυσίδων σε όλους τους πίνακες).
- Το μέγιστο μήκος των αλυσίδων (μέγιστο μήκος οποιασδήποτε αλυσίδας σε οποιονδήποτε πίνακα κατακερματισμού).
- Το μέσο πλήθος των αλυσίδων που είναι κενές (άθροισμα πλήθους αλυσίδων που είναι κενές σε κάθε πίνακα κατακερματισμού διά του πλήθους των πινάκων κατακερματισμού).

Τέλος, αν χρησιμοποιούνται πίνακες κατακερματισμού που ακολουθούν τη μέθοδο του ταξινομημένου διπλού κατακερματισμού θα πρέπει να τυπώνονται οι εξής πληροφορίες:

- Το ποσοστό των κλειδιών στον πίνακα κατακερματισμού που βρίσκονται στην αρχική τους θέση.
- Το μέγιστο πλήθος κλειδιών που έχουν την ίδια τιμή κατακερματισμού.
- Το πλήθος των κενών θέσεων του πίνακα κατακερματισμού.

Όταν χρησιμοποιείται κατακερματισμός που ακολουθεί τη μέθοδο του ταξινομημένου κατακερματισμού, επιλέξτε τρεις διαφορετικές τιμές για τον παράγοντα φόρτωσης (π.χ., a = 0.3, a = 0.6 και a = 0.8) και κάθε φορά που το a γίνεται ίσο με μια συγκεκριμένη από αυτές τις τιμές για κάποιο πίνακα κατακερματισμού

τυπώστε το μέσο πλήθος προσπελάσεων για επιτυχή αναζήτηση (όπου δεν προσμετρούνται οι προσπελάσεις μνήμης που απαιτούνται για την εύρεση του κατάλληλου κόμβου). Πιο συγκεκριμένα, για κάθε πίνακα κατακερματισμού θα πρέπει να διατηρούνται διάφοροι μετρητές, όπως το μέγιστο πλήθος στοιχείων που είναι αποθηκευμένα στον πίνακα, το συνολικό πλήθος προσπελάσεων που έχουν πραγματοποιηθεί για την αναζήτηση στοιχείων στον πίνακα και το συνολικό πλήθος αναζητήσεων που έχουν πραγματοποιηθεί (ώστε να είναι δυνατός ο υπολογισμός του μέσου πλήθους προσπελάσεων για τον πίνακα αυτόν). Όταν ο παράγοντας φόρτου για τον εκάστοτε πίνακα πάρει μια από τις τρεις τιμές που έχετε επιλέξει, θα πρέπει να αποθηκεύονται κατάλληλα δεδομένα ώστε να μπορεί να γίνει τελικά η εκτύπωση που συζητείται στη συνέχεια. Τα αποτελέσματα να δοθούν σε μορφή πίνακα, όπου να φαίνονται, για κάθε τιμή του a, οι τιμές του μέσου πλήθους προσπελάσεων που προσμετρήθηκαν, οι τιμές που προκύπτουν από τη θεωρία και οι ποσοστιαίες αποκλίσεις.

Bonus: Υλοποίηση Ξένων Συνόλων με Ένωση [30%]

Ζητείται να γίνει η εξής τροποποίηση στη δομή δεδομένων των αρχείων ενός κόμβου. Το πεδίο tp κάθε στοιχείου της δομής δεν θα πρέπει να είναι ένας ακέραιος αλλά ένας δείκτης σε κάποιο κατάλληλο κόμβο ενός πάνω δένδρου (που αντιστοιχεί σε αυτό το αρχείο). Έτσι, για την αποθήκευση των αρχείων ενός κόμβου χρησιμοποιείται επιπρόσθετα του πίνακα κατακερματισμού και ένα δάσος από τρία πάνω δένδρα, ένα για κάθε τύπο αρχείων.

Όπως έχει αναφερθεί, κατά την αποχώρηση ενός κόμβου από το σύστημα, ο κόμβος αφήνει τα αρχεία του στο επόμενό του στο δακτύλιο. Τότε θα πρέπει να εφαρμοσθεί η λειτουργία της ένωσης μεταξύ των κατάλληλων πάνω δένδρων των δύο κόμβων.

Όταν ένας κόμβος v εισάγεται στο σύστημα, κάποια από τα αρχεία του επομένου του (έστω u) στον δακτύλιο θα πρέπει να αποθηκευθούν στο νέο-εισαχθέντα κόμβο. Τα αρχεία αυτά διαγράφονται από τις δομές δεδομένων του u και εισάγονται σε αυτές του v. Δεδομένου ότι τα πάνω δένδρα δεν υποστηρίζουν αποδοτικά την λειτουργία της διαγραφής, τα πάνω δένδρα του u καταργούνται (ελευθερώνοντας όλους τους κόμβους τους) και όταν γίνει ο διαχωρισμός των κλειδιών μεταξύ των κόμβων u και v, τα πάνω δένδρα του u (επίπροσθετα αυτών του v) σχηματίζονται από την αρχή (ξεκινώντας από τόσα μονοσύνολα όσα και τα αρχεία που απομένουν στον u και συνενώνοντας κατάλληλα αυτά ώστε να προκύψουν τα τελικά δένδρα).

Επίσης, θα πρέπει να προστεθεί ένα ακόμη γεγονός όπως περιγράφεται στη συνέχεια:

- T <id> <tp>: Γεγονός τύπου Type το οποίο σηματοδοτεί την εκτύπωση εκείνων των αρχείων που αποθηκεύονται στον κόμβο με αναγνωριστικό <id>, των οποίων ο τύπος είναι <tp>.

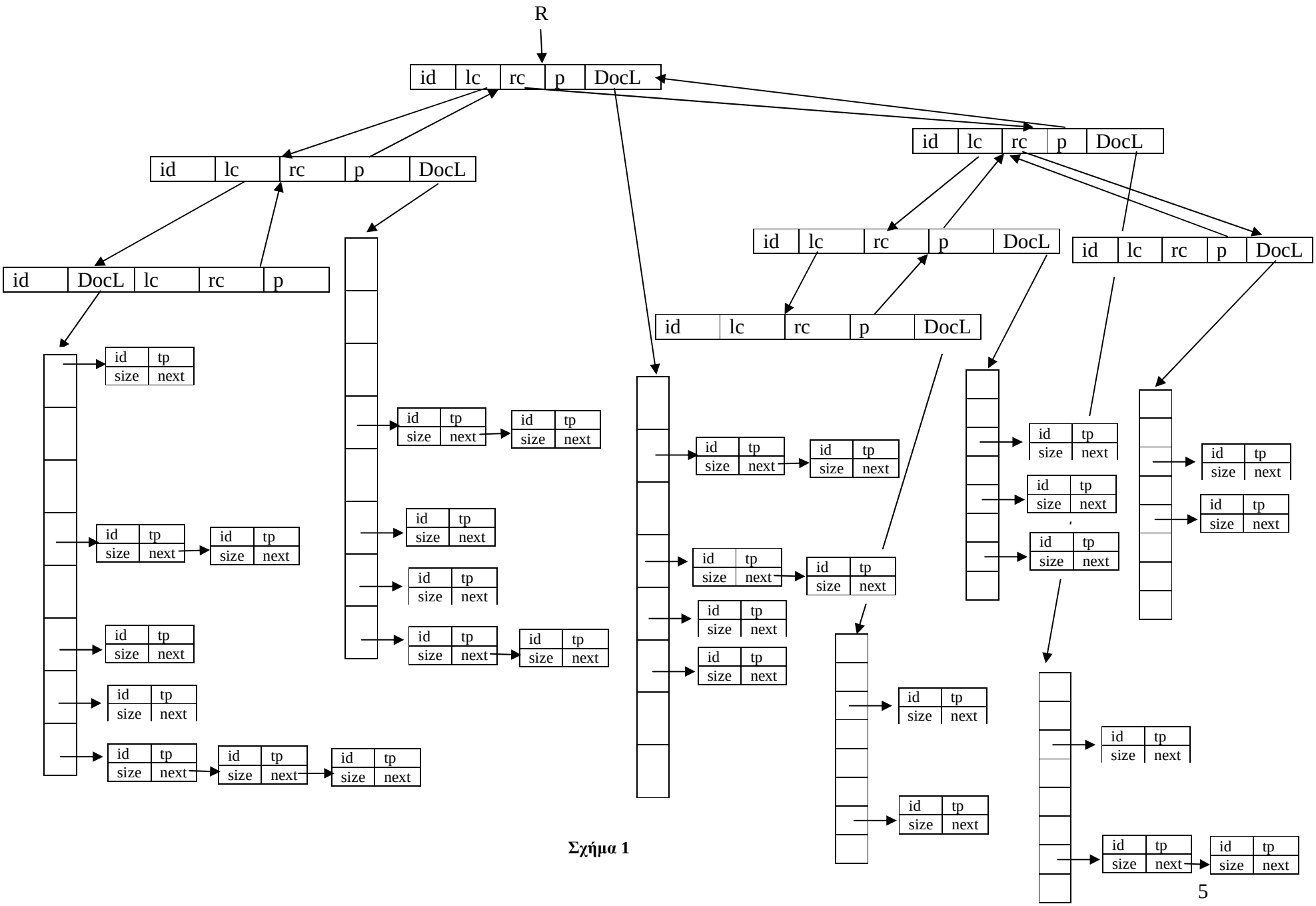
  Μετά το πέρας της εκτέλεσης ενός τέτοιου γεγονότος το πρόγραμμα θα πρέπει να τυπώνει την ακόλουθη πληροφορία:

  T <id> <tp> DONE
  DOCLIST = $<fd_1>$, $<fd_2>$, …, $<fd_k>$,

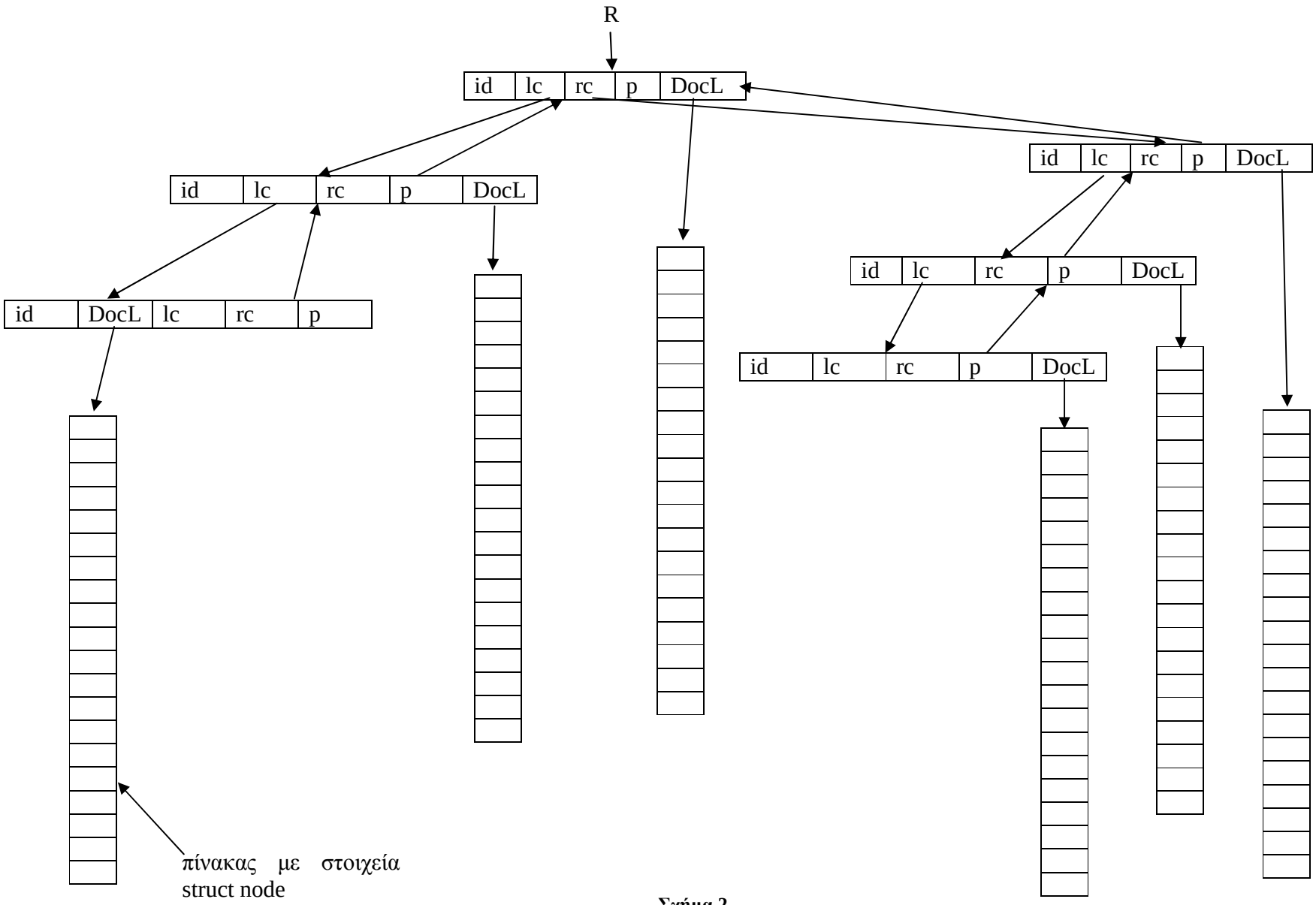
όπου k είναι το πλήθος των αρχείων τύπου <tp> που είναι αποθηκευμένα στον κόμβο με αναγνωριστικό <id> και για κάθε j, $1 \leq j \leq k$, $<fd_j>$ είναι το αναγνωριστικό ενός τέτοιου αρχείου.

Οδηγίες και Συμβουλές για την Ομαλή Διεκπεραίωση της Εργασίας

Όπως και με τα προηγούμενα μέρη της εργασίας, το μέρος αυτό θα πρέπει να πραγματοποιηθεί σε βήματα. Ο κάθε φοιτητής είναι πλέον ώριμος να αποφασίσει ποια βήματα θα απλοποιήσουν την εργασία του και θα οδηγήσουν στην ομαλή διεξαγωγή της.



Σχήμα 1


R
id
lc
rc
p
DocL
id
lc
rc
p
DocL
id
lc
rc
p
DocL
id
DocL
lc
rc
p
id
lc
rc
p
DocL
id
lc
rc
p
DocL
πίνακας με στοιχεία struct node

Σχήμα 2